Agent-based Framework for Modelling and Simulation of Resource Management in Smart Self-Sustainable Human Settlements

Tomičić, Igor

DOCTORAL THESIS INFORMATION

I. AUTHOR

Name and surname	Igor Tomičić
Place and date of birth	Karlovac, May 5th 1983
Faculty name and graduation date for level VII/I	Faculty of Organization and Informatics, September 17th 2007
Faculty name and graduation date VII/II	-
Current employment	University of Zagreb Faculty of Organization and Informatics

II. DOCTORAL THESIS

Title	Agent-based Framework for Modelling and Simulation of Resource Management in Smart Self-Sustainable Human Settlements
Number of pages, figures, tables, appendixes, bibliographic information	206 pages, 46 figures, 20 tables, 15 listings, 2 appendixes, 237 bibliographic information
Scientific area and field in which the title has been awarded	Social Sciences, Information and Communication Sciences
Mentors	Asst. prof. Markus Schatten Prof. Pietro Terna
Faculty where the thesis was defended	Faculty of Organization and Informatics, Varaždin
Mark and ordinal number	

III. GRADE AND DEFENCE

Date of doctoral thesis topic acceptance	June 16th 2014
Date of doctoral thesis submission	May 18th 2015
Date of doctoral thesis positive grade	March 21rd 2016
Grading committee members	Assoc.prof. Kornelije Rabuzin Assoc.prof. Robert Fabac Prof. Tarzan Legović Prof. Pietro Terna Asst. prof. Markus Schatten
Date of doctoral thesis defence	May 16th 2016
Defence committee members	Assoc.prof. Kornelije Rabuzin Assoc.prof. Robert Fabac Prof. Mirko Maleković Prof. Pietro Terna Asst. prof. Markus Schatten
Date of promotion	

University of Zagreb

Faculty of Organization and Informatics

Igor Tomičić

AGENT-BASED FRAMEWORK FOR MODELLING AND SIMULATION OF RESOURCE MANAGEMENT IN SMART SELF-SUSTAINABLE HUMAN SETTLEMENTS

- DOCTORAL THESIS -

Thesis advisers:
Asst. prof. Markus Schatten
Prof. Pietro Terna

Varaždin, 2016.

Sveučilište u Zagrebu

Fakultet organizacije i informatike

Igor Tomičić

AGENTNO RAZVOJNO OKRUŽJE ZA MODELIRANJE I SIMULACIJU UPRAVLJANJA RESURSIMA U PAMETNIM SAMOODRŽIVIM LJUDSKIM NASELJIMA

- DOKTORSKI RAD -

Mentori:
Asst. prof. Markus Schatten
Prof. Pietro Terna

Varaždin, 2016.

*"**Freedom** is the greatest fruit of self-sufficiency."*

- Epicurus

to Branka

and my two families

– for teaching me the true meaning

of unconditional support.

ABSTRACT

A human settlement may exist in an environment which does not allow the use of central resource distribution systems such as national electrical grid, municipal water supply, etc., or where the use of such systems is not preferable by its inhabitants. In such scenarios, there is a need for managing resources that are being produced, stored, and consumed, within the boundaries of such settlements, with a goal to maintain the self-sustainability of such settlement. Settlements are hereafter considered self-sustainable if sufficient resource quantities are available in the observed time period, with regard to defined comfort levels and observed resource type.

Considering existing work in the fields of smart cities, self-sustainable settlements, smart grids, Internet of things and existing intelligent agent infrastructures tackling renewable resource management, new agent classes, behaviours and protocols are proposed. A central part of the dissertation work includes developing specialized agent classes and behaviours that would allow to model and simulate the dynamics of a local resource production, storage and consumption in order to pursue and evaluate self-sustainability of the settlement.

Such framework could be used by a modeler either to design and simulate a new self-sustainable system, or to evaluate and analyze an existing system, which could lead to considerable improvements in the design of a settlement and its resource production, consumption and storage elements, in regard to the context of self-sustainability.

Keywords: self-sustainability, multi-agent system, MAS, agents, methods, simulation, modeling, framework

EXTENDED ABSTRACT

The aim of the study is to model and simulate self-sustainable human settlements in the newly developed smart self-sustainable human settlement framework (abbrv. SSSHS), in order to prolong the self-sustainability of the settlement. To this end, the following goals have been identified:

1. To develop new agent classes, behaviours and protocols which would be specialized for modelling and simulation of resource management in smart self-sustainable human settlements.
2. To integrate newly developed SSSHS agent classes, behaviours and protocols into a functional modeling and simulation framework.
3. To develop nontrivially complex test-bed scenarios for SSSHS resource management. Possible scenarios fall within three categories: permaculture, space colony, space travel.

The hypothesis of the dissertation is the following:

H1. The agent-based framework for modelling and simulation of smart resource management in human settlements will prolong the time interval of self-sustainability when used on the developed test-bed scenarios.

The 1st chapter introduces the research and places it into context. It presents the most concise research elements: research goals and research hypothesis.

Chapter 2 elaborates the research in more detail, starting from the initial conceptual framework, towards building the framework, implementation of the framework, with implementation and research methods' specifics.

Chapter 3 presents a literature overview covering research areas such as sustainable development, self-sustainable human settlements, agents and multi-agent systems, intelligent buildings and smart cities, eco feedback, and load management.

Chapter 4 presents the framework implementation. Following the adapted workflow of the MaSE development methodology, it first identifies system requirements, from which the system goals are derived. Goals are then used for the creation of agent roles, and roles are used for the definition of classes.

The overall goal of the SSSHS framework is stated as follows:

> *After persistently trying to make the modeled system self-sustainable, inform the modeler if the modeled system is self-sustainable; if the modeled system is not self-sustainable, refer the modeler to the relevant problems that beset the self-sustainability.*

The same chapter elaborates used resource consumption models and self-sustainability mechanisms. These mechanisms were derived from the reviewed literature on the relevant research areas, from the informal interviews with three experts on self-sustainability, from the real-world resource distribution mechanisms used by the gas and electric companies, and from the knowledge of the operation of a number of widely used consumption units, personal experiences and analysis.

Chapter 5 explains the usage of the SSSHS framework in modeling and simulating specific systems. User input (used for modeling and additional framework settings) and reporting variables (used for the simulation analysis) are presented in detail.

Chapter 6 present the three developed test-bed scenarios, simulation results and discussion. The scenarios were derived from the observed environment (an analysis of the eco village in Sisak-Moslavina County, Croatia, Mars One mission technical feasibility, and NASA research into human stasis), and preliminary analyzed via spreadsheets and simple mathematical formulas. Such scenarios were then modeled in the SSSHS framework, and simulated with the same parameters, but with the self-sustainability mechanisms active during the simulation run. The results showed that the SSSHS framework was able to prolong the self-sustainability in every simulated scenario, which is consistent with the research Hypothesis H1.

Chapter 7 concludes the research, argues the results of the simulation runs, and proposes the plans for further research.

Keywords: self-sustainability, human settlements, resources, multi-agent system, MAS, agents, methods, simulation, modeling, framework

SAŽETAK

Cilj istraživanja je modelirati i simulirati samoodrživa ljudska naselja u novorazvijenom razvojnom okružju za modeliranje i simulaciju upravljanja resursima u pametnim samoodrživim ljudskim naseljima temeljenom na agentima, sa svrhom produljenja samoodrživosti naselja. U tom kontekstu, identificirani su sljedeći istraživački ciljevi:

1. Razviti nove klase agenata, ponašanja i protokola specijaliziranih za modeliranje i simulacije upravljanja resursima u pametnim samoodrživim ljudskim naseljima.
2. Integrirati novo razvijene klase agenata, ponašanja i protokola u funkcionalno razvojno okružje za modeliranje i simulaciju.
3. Razviti ne-trivijalno kompleksne scenarije za modeliranje i simulaciju upravljanja resursima u pametnim samoodrživim ljudskim naseljima. Scenariji će biti razvijeni unutar tri kategorije: permakulturno eko-selo, vanzemaljska kolonija, svemirsko putovanje.

Hipoteza disertacije je sljedeća:

H1. Razvojno okruženje za modeliranje i simulaciju upravljanja resursima u pametnim samoodrživim ljudskim naseljima temeljeno na agentima produžiti će vremenski period samoodrživosti prilikom korištenja u razvijenim testnim scenarijima.

Prvo poglavlje je uvodno i elaborira kontekst istraživanja u konciznom obliku, predstavljajući istraživačke ciljeve i hipotezu.

Poglavlje 2 detaljizira istraživanje, od inicijalnog koncepta okružja, prema gradnji okružja i implementaciji, dajući uvid u detalje implementacije i korištene istraživačke metode.

Poglavlje 3 pruža pregled literature područja relevantnih za istraživanje, kao što su održivi razvoj, samoodrživa ljudska naselja, agenti i više-agentni sustavi, inteligentne zgrade, pametni

gradovi, eko-povratne informacije, upravljanje vršnim opterećenjem, te drugih povezanih područja.

Poglavlje 4 prezentira implementaciju razvojnog okružja. Slijedeći prilagođeni tok rada MaSE metodologije, u prvim koracima identificiraju se zahtjevi sustava, prema kojima se deriviraju ciljevi sustava. Ciljevi se nakon toga koriste u kreiranju agentnih uloga, a uloge se potom koriste za definiciju klasa agenata.

Globalni cilj razvojnog okruženja je sljedeći:

> *Nakon izvršenih aktivnih napora usmjerenih u svrhu ostvarenja samoodrživosti modeliranog sustava, informiraj korisnika o tome da li je sustav samoodrživ; ukoliko modelirani sustav nije samoodrživ, usmjeri korisnika na relevantne probleme koji sprječavaju samoodrživost.*

Isto poglavlje elaborira modele konzumacije resursa i mehanizme samoodrživosti. Mehanizmi samoodrživosti izvedeni su iz literature povezanih i relevantnih područja istraživanja, iz intervjua provedenih sa ekspertima za samoodrživost, iz stvarnih mehanizama koje koriste primjerice poduzeća za energetiku, te iz poznavanja načina rada velikog broja jedinica za potrošnju energije, analize i vlastitih iskustava.

Poglavlje 5 elaborira način korištenja razvojnog okružja prilikom modeliranja i simulacija specifičnih sustava. Pojašnjene se ulazne i izvještajne varijable koje korisnik koristi za modeliranje, prilagodbu i analizu sustava.

Poglavlje 6 opisuje razvijene testne scenarije, rezultate simulacija nad tim scenarijima, te analizu i diskusiju. Scenariji su inicijalno simulirani bez korištenja mehanizama za samoodrživost, koristeći jednostavne matematičke formule u alokaciji resursa. Nakon toga, scenariji su simulirani nad identičnim ulaznim podacima koristeći mehanizme za samoodrživost razvojnog okružja. Rezultati provedenih simulacija pokazali su da je razvojno okružje produžilo trajanje samoodrživosti u svakom od razvijenih scenarija, što je konzistentno sa istraživačkom hipotezom H1.

U poglavlju 7 zaključuje se istraživanje, diskutira o rezultatima simulacije, te se predlažu planovi za daljnje istraživanje.

Ključne riječi: samoodrživost, ljudska naselja, resursi, više-agentni sustav, VAS, agenti, metode, simulacija, modeliranje, razvojno okruženje

TABLE OF CONTENTS

List of Figures

List of Tables

Listings

1. Introduction

Resource production in a self-sustainable settlement is highly dependent on local environment features, and resource consumption is bound to various types of resource consumers, each of which have certain characteristics that directly influence the system's production/consumption dynamics (operating hours, capacities, modes of operation, possible deviations, etc.). The main goal of the Smart Self-Sustainable Human Settlement (SSSHS) framework, which is to be developed in this thesis, is to facilitate an infrastructure for smart resource allocation and management in order to cover overall needs of the system for a specific resource, in a defined period of time.

Local resource production is intermittent [1-2], and the challenge emerges to provide resources on the exact time the consumers need them. To successfully meet this challenge, resource production, storage and consumption should be managed to enable sufficient quantities of resources to all consumers within the system upon requests.

Due to the complexity of such systems there is a need for the development of tools to allow for modelling, simulation and evaluation. A broad literature review has shown that currently such specific simulation infrastructure to support SSSHS does not exist, and thus a development of an agent-based framework for modelling and simulating resource management within SSSHS is proposed and executed within this dissertation. This infrastructure includes new classes and behaviours of agents specialized for modelling local/renewable resource management and, as such, actively tries to maintain the self-sustainability of the settlement for a given resource in the simulation run. Active maintenance of the self-sustainability implies the use of adaptive resource storage, negotiating and consumption policies.

Because of the complex nature of the production/consumption dynamics within the settlement, there is a need for autonomous and flexible actions regarding the resource management of such system. An agent-based modelling (ABM) approach provides an appropriate feasible solution for modelling and simulating such systems.

The SSSHS framework assumes the use of physical network infrastructure which would facilitate both data and resource exchange between individual dwelling units inside the self-sustainable human settlement. Within such infrastructure, resource allocation and management would become automated and efficient in terms of maintaining both the individual comfort levels, and the overall property of self-sustainability of the settlement. Such scenario would facilitate the efficient use of renewable resources with multiple benefits argued in section 3.1.
Interested communities would be able to freely use this framework in the future. Under an open source license, it would be available for individual customizations and adaptations for specific infrastructures. From a scientific perspective, it is expected that the SSSHS framework would give a momentum to the research efforts in using computer science techniques in facilitating the development of self-sustainable, resilient communities.

1.1 Defining Self-Sustainability in the Research Context

Apart from the definitions given in the literature overview in chapter 3., in the context of this thesis a system will be considered self-sustainable for a given resource, if its demands for that resource can be met with the resource quantities produced within its boundaries, and within the defined time period. Such system won't require external resources, but exclusively those (restricted) resources which are produced or gathered within the same system, from available natural sources. If resource demands within a self-sustainable system can't be met with its existing production capacities, the system is considered non self-sustainable, and thus is unable to function properly off the grid.

Advances in the areas such as networking, embedded systems, and computing, are allowing to design a model of mutually interconnected devices in an infrastructure which supports their dynamic networking, collaboration, and decision making. These advances and their resultant technologies could provide feasible foundations for building a real-world smart self-sustainable human settlement in the future, governed by the similar principles as described in this work.

1.2 Research Goals

In this thesis three main research goals have been set, which summarize the purpose of this research in a concise and focused form:

1. To develop new agent classes, behaviours and protocols specialized for modelling and simulation of resource management in smart self-sustainable human settlements.

As an extensive literature overview (presented in chapter 3) showed, currently classes, behaviours and protocols which would facilitate modeling and simulation of resource management in a smart self-sustainable human settlement do not exist, nor is there a considerable momentum of research in this area which would provide a ground for further development. Due to such an existing context, the development of such classes, behaviours and protocols from the ground up is herein considered to be an optimal solution.

2. To integrate newly developed SSSHS agent classes, behaviours and protocols into a functional modeling and simulation framework.

The SSSHS framework would facilitate modeling and simulation of self-sustainable human settlements. It would be possible to model either an existing settlements and to assess its properties regarding self-sustainability, or to conceptualize and model a new settlement and optimizing its parameters before its assumed physical implementation.

3. To develop nontrivially complex test-bed scenarios for SSSHS resource management. Possible scenarios fall within three categories: permaculture, space colony, space travel.

A test-bed is generally defined as an environment, or a platform, that is created for the purpose of conducting transparent and replicable testing of scientific theories and/or new technologies. Test-bed scenarios within permaculture category will be based on a real-life case of self-sustainable community and consulted with local self-sustainability experts in order to create a realistic simulation scenario. Data will be collected directly on the site, via several interviews with the domain experts, as well as from the government agencies, regarding past weather data and related

statistics. A detailed analysis of the settlement's capabilities, politics and dynamics regarding resource production, consumption and storage will be performed.

The test-bed scenario within the space colony category will be mainly based on the Mars One mission, a mission with the goal to establish a human settlement on Mars, and parameters relevant to the SSSHS framework will be partially derived from the Mars One technical feasibility paper [224].

Space travel will include a hypothetical scenario where a fleet of five independent and mutually networked space crafts are traveling to Mars in a total duration of 180 days. Parameters relevant to the SSSHS framework will be partially derived from the study conducted by the aerospace engineering organization SpaceWorks Enterprises [226], and the Mars One technical feasibility paper [224].

1.3 Research Hypothesis

The research hypothesis is focused on the defined term "self-sustainability" in the context of human settlements. The main goal of the developed simulation framework is to prolong the time interval of the human settlement's self-sustainability in regard to the simulated resource type, by using newly developed agent classes, behaviours and protocols, specifically designed for this purpose.

The research hypothesis is as follows.

H1. The agent-based framework for modelling and simulation of smart resource management in human settlements will prolong the time interval of self-sustainability when used on the developed test-bed scenarios.

Four scenarios will be modeled and simulated in the developed SSSHS framework which will facilitate the testing of hypothesis H1.

2. Research Plan

2.1 Initial Conceptual Framework

Based on a state-of-the-art literature overview and on expert opinions and guidelines derived from number of informal interviews with individuals from self-sustainable communities, a set of initial problems and demands on resource production, storage and consumption dynamics within a generic human settlement will be set, resulting in a conceptual model of the SSSHS resource management. A narrative and graphical description of the main components, their interactions and behaviours, the environment, and relevant input data within the dynamics of the resource management in human settlement, space colony and space travel, will be presented.

This conceptual model will be used as a main reference in building the framework. The system demands will gravitate around the central problem of resource allocation dynamics, and around the agents (intelligent systems) who wish to maximize the benefits obtained from a restricted resource.

Individual agents would aim to accomplish the common goal on a macro level, which would also be projected into their micro levels: maintaining the self-sustainability of the system regarding the simulated resource.

2.2 Building the Framework

Identified demands and features of the conceptual model will help to define agent roles, and agent roles will be mapped to agent classes. Agent classes will consist of attributes and methods (by which attributes are manipulated) specialized for dealing with identified problems within the specific domain of resource management in human settlements.

Appropriate communication and behaviour policies between agents will be defined, presented, and implemented in the framework. The environment in which the agents operate will be identified in terms of input data it provides to agents, possibly influencing their behaviours.

Core part of the proposed framework will consist of new agent classes, their behaviours and interactions, specialized for the SSSHS resource management processes.

2.3 Implementation

New agent classes, behaviours and protocols will be implemented into a functional modeling and simulation framework. This framework would accept user input in a modeling process, as well as in specifying the simulation parameters.

In the last phase the developed application will be tested on four scenarios in order to provide a necessary feedback on development. These scenarios will facilitate the testing of the research hypothesis H1.

2.4 Research Methods

In the conceptual phase of the development, general methods such as analysis, description, explanation, generalization, classification, compilation, comparison, and synthesis will be used.

Building the conceptual models will include the use of UML (Unified Modelling Language) object modelling language. UML is commonly adopted to support agent-based models, although it does not have a formal semantic and it does not clearly address the agent theory.
An agent-based approach has been selected because of several key system properties, which according to [3] and [4] point to the appropriateness of such approach:

1. **The environment is open, or at least highly dynamic, uncertain, or complex**. Environment in self-sustainable human settlements has a direct influence on the resource production and consumption dynamics, and is highly dependable on various factors including but not limited to geological, meteorological, architectural, and human behaviour, which are highly dynamic and often uncertain. In these kind of environments, "*systems capable of flexible autonomous actions are often the only solution*" [5, pp. 184].

2. **Agents are a natural metaphor**. Environments such as human settlements, composed of many resource production/consumption units, are easily modelled as a network of agents which communicate, cooperate, and negotiate with each other in order to achieve a predefined goal - maintaining the self-sustainability of the settlement by intelligently and flexibly sharing and managing resource, all in behalf of its owners – human residents.

3. **Distribution of data, control or expertise**. Each unit in a settlement has its own autonomous agent subsystem and knowledge bases, which tries to maintain self-sustainability of the unit, and in the same time, user comfort and transparentness of its operation. Actions that are taking place inside each settlement unit are managed by an agent subsystem, which again works in a dynamic and uncertain environment. At certain events, a unit agent initiates a communication with other agents, distributing data and negotiating for resource transfer. According to Wooldridge, in such situations a centralized solution would be thus either extremely difficult, or even impossible [5].

3. Research Context and Literature Overview

The following sections present an overview of the research areas identified as relevant for the context of this research, such as sustainable development, agent based modelling and multi-agent systems, intelligent buildings, eco feedback, load balancing, Internet of things and others. These sections also provide an insight into the basic notions used in the SSSHS framework, such as self-sustainability, self-sustainable human settlements, resources, etc.

3.1 Sustainable Development and Self-Sustainability

The term "sustainable development" was articulated by the United Nations in 1987 in the report of the World Commission on Environment and Development (WCED), and was defined as "*development that meets the needs of the present without compromising the ability of future generations to meet their own needs*" [99, pp. 42].

In the year 1994, John Elkington has proposed the following three different aspects of sustainability [100]:

- environmental
- social
- economic

The main idea behind this "triple bottom line" concept (abbreviated as TBL or 3BL, often labeled as the "three pillars of sustainability") was that business activity can simultaneously deliver *financial*, *social*, and *environmental* benefits.

It was common for many people to consider sustainability as a synonym for energy-efficiency [101]. However, Scott [102] argues that sustainability is not only about energy efficiency, but also about user's comfort, well-being and cost savings.

When modelling, building, or merely considering aspects of a self-sustainable community, user comfort could be one of the key factors involving the consumption dynamics within the resource managing system.

Because many alternative and modified definitions have emerged since the initial definition (more than 140, [103]), the term “sustainable development” has acquired a broad meaning with no formal or generally agreed definition, and often implies a long-termed, strategic development, or organizing principle.

When summarizing all the definitions and descriptions together, one could conclude that the main focus of nearly all sustainability goals seem to be a fair distribution of resources [104].

Dictionary definitions offer more concise approach, and define the word “*sustainable*” as:

(a) “*able to be used without being completely used up or destroyed*”
(b) “*involving methods that do not completely use up or destroy natural resources*”
(c) “*able to last or continue for a long time*” [105]
(d) “*able to continue over a period of time*”
(e) “*causing little or no damage to the environment and therefore able to continue for a long time*” [105]

Dictionary definitions of the term “*self-sustaining*” are presented as follows:

1. ”*maintaining or able to maintain oneself or itself by independent effort <a self–sustaining community>*” [107]
2. "*Able to sustain oneself or itself independently.*” [108]

By reviewing these dictionary definitions, the distinction between those two terms can be identified as an “independence” factor, which is an integral part of the definition of the term “self-sustaining”, while in the same time it is absent in the definition of the word “sustaining”.

Self-sustainable communities imply a cyclical processes in terms of creating and using resources, as opposed to the "classical" linear processes established in most urban settlements. Karl-Henrik Robèrt, founder of the Natural Step Initiative argues that "*the only processes that we can rely on indefinitely are cyclical; all linear processes must eventually come to an end*" [109]. Robèrt observes that our society is continuously processing natural resources in a linear direction, which in time reaches an end, thus being not sustainable. These processes are unidirectional, extracting the resources from the nature and rapidly transforming them into "*useless garbage, some of which is obvious to the naked eye, but most of which escapes awareness. The smaller portion can be seen in garbage dumps and other visible waste. By far the larger portion can be thought of as 'molecular garbage' - consisting of the vast quantities of tiny particles that are daily spewed out into the earth's air, water and soil.*" [109]

There are several relevant advantages in using self-sustainable systems opposed to using linear consumeristic models, as described hereafter:

- A self-sustainable system is not dependent on availability, capacity or price fluctuations of the central resource distribution system, because self-sustainable system is not connected to and does not use resources provided by the central distribution systems.
- A self-sustainable system minimizes resource losses due to transportation, considering the small physical scale and compact nature of the self-sustainable system, in comparison with physical resource transportation lengths from central distribution systems.
- Self-sustainable systems assume the usage of resources extracted from local renewable resources, which has numerous advantages over using resources derived from fossil fuels, considered to be the root problem for global warming [110].
- A self-sustainable system supports and educates ideas of resilient communities, transition movement, localization and permaculture [111].
- Self-sustainability has a potential to help low-income families to achieve economic independence. [112-113]

As far as currently dominated linear model trends in energy supply and consumption are considered, numerous established organizations have concluded that such trends are unsustainable

[114-115]. These organizations include International Energy Agency [116] and European Union [116-119] for example.

Related to previously discussed trends, the rise of greenhouse gas emissions (mainly CO_2) is threatening the planet's global climate and chemistry, and the strategy of the European Union in the field of energy and climate change propose three commitments to be met by 2020: to reduce greenhouse gas emissions by at least 20%, to ensure that 20% of final energy consumption is met with renewable sources, and to raise energy efficiency by 20% [120].

During recent years, using renewable energy sources has gained significant interest. The final purpose of exploitation of renewable energy sources is to move towards increased energy sustainability, and eventually complete independence from fossil fuel energy [121].

3.2 Resources

"Resources" are considered as abstract values in the context of human settlements and for the framework's simulation needs (abstraction provides the framework with the ability to simulate various types of resources without making any internal changes), and there is no need to elaborate their specifics, ontology, their sources, usages, in great detail. However, in order to achieve basic understanding about motivation, fundamentals and potential usage scenarios of the proposed model, some of the potentially simulated resources are explained hereafter. Existing research papers on the uses of specific resources will be presented, and their results will be used in the development of framework's test scenarios.

Since the main focus of the work are human settlements, mainly considered resources are defined from a human standpoint as "*anything that we can obtain from the environment to meet our need and wants.*" [9, pp. 11]

Some of the resources are directly available for use (such as solar energy, edible wild plants, fertile soil, etc.), while some of them become useful only with certain effort and by using certain technological solutions (such as underground water, iron, petroleum, etc.). Also, resources vary in

terms of how quickly nature can replenish them after they are used; a resource that takes "*anywhere from several days to several hundred years to be replenished through natural processes is a renewable resource, as long as we do not use it up faster than nature can renew it*" [9, pp. 10].

3.2.1 Water

Water has a significant impact to the overall resource consumption in an average household. It is a resource vitally important for maintaining human health, producing food and maintaining aquatic ecosystems [10]. Consumed quantities of water are usually referred to as liters per person per day (from now on, in this research referring to as acronym l/p/d).

Water is mainly used in the domestic, industrial and agricultural contexts, whereas domestic use is primarily focused in the context of this research - for example, water used in sanitation, bathing, drinking, cooking, heat exchange, home-scale gardening, recreation, etc.

In order to establish a minimum quality of life to humans, basic water requirements in a household have been estimated by approximately 50 liters per person per day for the needs of drinking (5 l/p/d), food preparation (10 l/p/d), sanitation (20 l/p/d) and bathing (15 l/p/d); here excluding water for gardening [10]. Similar parameters are valid for space-related scenarios. Water requirements for food growing depends on numerous factors like regional climate conditions, irrigation and food processing technologies, cultural preferences, minimum caloric requirements in diet, etc. [10].

The following known possible sources of water in a dwelling network could be utilized [11]:

1. Surface waters (rivers, lakes, wetlands, artificial reservoirs). Source of these waters is precipitation (l/m^2).
2. Ground waters. Source: seepage from surface water.
3. Desalination. Source: seawater, etc.

3.2.2 Electricity

Electricity has several definitions differentiated by distinct points of view. As such, it can be defined as a form of energy derived from the existence of charged particles; it may exist dynamically in the form of a current, or statically as a charge accumulation [12]. Also, electricity from the user perspective, which is the most significant perspective in the context of this research, may be defined as a supply of electric current to human dwellings in order to provide for lightning, appliances, heating, etc. In linear, "traditional" one-way systems, the supply of electricity originates in a supplier's central distribution system, and travels via electrical grid system to numerous consumers, usually dispersed in a large land area.

When considering self-sustainable human settlements, there is usually no central production system and distribution from a remote location. Instead, electricity may be generated on-site via various alternative sources, such as [13]:

- Solar panels
- Used edible oils
- Wind turbines
- Dams
- Methane gas
- Biodiesel
- Waves, currents, tides (turbines)
- Salinity and temperature gradients (thermodynamical engines)
- Atmospheric electricity, etc.

A kilowatt-hour (symbolized kWh) is commonly used as a standard unit of energy.

3.2.3 Thermal Energy

Thermal energy is the internal energy of a system or a substance that is directly related to its temperature, i.e. it comes from the heat. The heat is generated by the movement or vibration of

particles within an object. Thermal energy storage in general has been an important topic in research for the last two decades [14].

Thermal energy can be used for heating the dwellings or vehicles, cooking food, generating electricity, etc. Potential sources of thermal energy are the following [13]:

- Fossil fuels
- Biomass (organic materials such as forest debris, manure, compost piles, scrap lumber, etc.)
- Solar systems (for example solar water heaters, solar cookers, etc.)
- Electricity
- Mechanical forces
- Geothermal sources

Commonly used units for thermal energy include joule (J, International System of Units), British thermal unit (BTU), and calorie.

3.2.4 Food

In the context of this research, food can be defined as any nutritious substance (edible or potable) consisting essentially of protein, carbohydrate and fat, used in the body of an organism to sustain life; facilitate growth, repair of tissue, and vital processes; and to provide energy [15].

As food is also an abstraction in the context of the simulation framework to be developed, there is no need to further classify or detail the subject of food - its types, cultivation, calorific values, and similar - this would be a context for a modeler of the specific system, with specific inhabitants, their preferences, surroundings, and similar factors.

As observed in [16], cities have become dependent on large amounts of food being transported in from outside the land area they occupy. London is given as an example; its surface area measures approximately 160000 ha, and contains about 12% of Britain’s population. In regard to this,

London requires the equivalent of 40% of Britain’s entire productive land for its food - which, of course, is being transported from cultivation sites all around the globe.

The food distribution systems which depend on motorized transport are having a significant negative impact on air pollution, fossil fuel use, and damage to wildlife habitats - which is mostly resulting from the road building [17]

One of the possible solutions to negative environmental impacts resulting from the food distribution and transporting processes is to grow food locally, which is intrinsic to the self-sustainable communities. Alternative food networks (AFNs) could provide a space for different food distribution approach; as stated in [18], AFNs are defined in four major ways:

1) Shorter distances between producers and consumers. It is assumed that food producers (farmers) grow food in physical proximity to their buyers
2) Smaller farm size and scale; organic or holistic farming methods.
3) Existence of food purchasing venues such as food cooperatives, farmers markets, and CSA (Community Supported Agriculture) and local food-to-school linkages
4) Commitment to the social, economic and environmental dimensions of sustainable food production, distribution and consumption.

These characteristics describe a holistic view on one food production-distribution-consumption system, and could be a subject of interest and a general guiding thread for self-sustainable communities.

3.3 Choosing Resources for Test-Bed Scenarios

“Water is the driver of Nature.”
- Leonardo da Vinci

Considering previously described resources and their potential for the use in the simulation scenarios, water has been chosen as one of the candidates for the scenario development. There are several reasons arguing the choice:

1. A real-world case in the self-sustainable human settlement involved a shortage of water in the settlement, and a number of parameters involving this case is obtainable from the source. This would provide adequate realistic basis for the development of a test-bed involving the permaculture scenario.
2. There is a significant number of research and guides involving both the use and collection of water, with relevant and usable information. This information includes for example water usage statistics in various use-cases, rainfall collection formulas, etc.
3. It is possible to obtain relevant meteorological (precipitation) data from the appointed government agencies, for the purpose of this research. This fact implies the use of real data, absent of approximations, interpolations, or mere guesses.
4. In space-related manned missions, water is used for drinking, hygiene, food preparation, and represents an indispensable resource integrated in life support systems [19].

The electrical energy was also chosen as a candidate for the scenario development, produced by the photovoltaic solar panels and auxiliary generators. There are several reasons arguing this choice:

1. Scenarios involving electrical energy would be developed under the supervision of the expert for the off-grid systems, Loren Amelang (Appendix A), and with a partial reference to the existing eco-village settings, making the scenario as realistic as relevantly possible.

2. The possibility to model and simulate more than one resource type is considered scientifically curious and challenging for the SSSHS framework, and the scenario settings involving resource producers which consume another type of resource to produce its default type of resource are able to facilitate such a possibility.
3. It is possible to obtain relevant meteorological data (for example, solar irradiance) from the appointed agencies, and nominal values of the equipment from the manufacturers for the purposes of calculations relevant to this research, which would minimize the need of approximations and interpolations of needed data.

Therefore, the SSSHS framework will simulate the following resources (some of them simultaneously), using the appropriate agent models for representing physical storage units:

- Water (using liters)
- Electrical energy (using watt-hours and ampere-hours)
- Propane gas (using liters)

3.4 Self-Sustainable Human Settlements

Implementing a self-sustainable system in human neighborhoods/settlements is considered more feasible than implementing a self-sustainable system for independent dwellings individually. Resource production capacity for an individual dwelling highly depends on numerous dynamic factors such as those deriving from the geographical position and orientation of individual houses, including a myriad of parameters of their local environment, and most importantly, on the complex behaviour of their residents.

Considering the feasibility of implementing a self-sustainable system in an individual dwelling, an illustrative scenario is offered here to further clarify the argument. A single resident of the house is considered, which produces heat exclusively from “manually” burning biomass. In this case, the capacity of the house for this specific resource (heat) production is limited to time periods when the resident is present at home. However, in order to retain some defined comfort level, this resource might be needed before the time the resident arrives at home, especially in the cold time periods. Such resource could be transported per request from one of the neighboring dwellings, assuming that some of those dwellings has a surplus of the needed resource in the moment of request.

From such a hypothetical viewpoint, a network of resource production/consumption units could be realistically more feasible for implementing a self-sustainable system, than it would be for a single house. For the given scenario, a dwelling is considered not self-sustainable in the event of loss of a resource. If the “loss” of the resource is defined as resource level dropping beneath some predefined comfort zone value, a dwelling would become not self-sustainable as soon as the ambient temperature drops beneath, for example, 18°C. Because the production of the heat directly depends on the availability of the house resident, it is likely that the house would not be self-sustainable in the cold time periods, because there would be no other methods of obtaining the needed resource. With the network of infrastructurally interconnected dwellings, resource transfers per requests could be available, raising the probability for a house (as a part of the system) to maintain the self-sustainability property.

In the reviewed literature, Blumendorf notes that self-sufficiency can "*hardly be achieved in a single household.*" [122, pp. 3]. Sustainable neighborhoods would achieve three broad sustainability goals: eco-efficiency, eco-equity and eco-effectiveness [123].

Interestingly, in about 400 BCE, Aristotle referred several times to a self-sufficiency concept in his work titled Politics:
"*For a household is more self-sufficient than an individual person is; and a community of a mass of people counts as a city only if it proves to be self-sufficient.*" [124, pp. 934]
"*A city is the community of families and villages in a complete and self-sufficient life. This sort of life, as we say; is a happy and fine life; hence we should suppose that a city aims at fine actions, not <merely> at living together.*" [124, pp. 344]

The proposed SSSHS framework considers human settlements, which are composed of two or more dwelling units (stationary, or mobile). Those dwelling units are assumed to be interconnected with a network facilitating both data and resource exchange, and thus can communicate with each other, and exchange physical resources.

3.5 Agents and Multi-Agent Systems

Because there is no universal and generally-agreed upon definition on the term "agent" [20], numerous and various definitions of an agent exists in a literature; some of them are stated as follows:

- "*An agent is a computer system that is situated in some environment, and is capable of independent (autonomous) action in this environment on behalf of its user or owner, in order to meet its delegated objectives.*" [5, pp. 15]

- "*An agent is a logic program which continuously executes observe-think-act cycle.*" [21, pp. 12]

- Agents are systems that are capable of independent decisions upon their next steps; an intelligent agent is agent that is autonomous and flexible, where flexibility assumes reactivity, proactivity and the ability to interact with other agents. [20]

- "*An agent is a software entity with a well-known identity, state and behavior, with autonomy to somehow represent its user.*" [22, pp. 3]

- *"An agent is everything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors."* [126, pp. 31]

- "*An agent is an object with goals.*" [23, pp. 24]

- *"An autonomous agent is a system situated within and a part of an environment that senses that environments and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future."* [127, pp. 5]

- "*Agents are autonomous, persistent (software) components that perceive, reason, communicate and act in someone's favour, influencing its environment.*" [24, pp. 1].

- *"An agent is a system with the following properties*:
    - *It lives in an artificial world W*
    - *It has facilities to sense W and to manipulate W*
    - *It has a (at least partial representation) of W*
    - *It is goal-directed, and as a consequence it has the ability to plan its activities*
    - *It can communicate with other agents"* [128, pp. 2]

- Intelligent agents are entities that continuously execute three functions: perceiving the environment, taking actions in order to change the environment, and reasoning in order to interpret perceptions, solve problems, achieving conclusions, and determining actions. [25]

- "*An intelligent agent is an entity with mental state which is composed of belief and goals.*" [26, pp. 78-89]

According to their characteristics, there are numerous types of agents described in the literature. [27] summarizes three defined approaches for agent infrastructures:

- **Deliberative** Architectures – adequate when long-term planning and reasoning is essential.
- **Reactive** Architectures – adequate for agents situated in highly dynamic environments where quick answers are essential.
- **Hybrid** Architectures – adequate for agents situated in unknown and changing environments or when a reactive behavior is as needed as long-term planning.
- **Cognitive** arhitectures – architectures that propose computational processes that act like certain cognitive systems (most often, like a person).

[28] describes rational and reactive agents; **rational agents** have their goals and beliefs represented explicitely, and **reactive agents** do not possess their own goals or memory; [20] defines reactive agents as those which react on the state of the environment without examining the past events. Such agents are also called tropistic agents [29]; tropism is the tendency of animals or plants to react to certain stimulae. [30]

Proactive agents act autonomously and try to achieve their goals by executing a set of actions. [31]. A proactive agent engages in decision making and information gathering whenever possible [32], while a reactive agent waits until being asked or until it is absolutely necessary to gather information or make a decision [33]

Deliberative (or intentional) agent is the "*one that possesses an explicitly represented, symbolic model of the world, and in which decisions (for example about what actions to perform) are made via symbolic reasoning*" [34, pp. 42]. Deliberative agent's internal processes are more complex compared to those of reactive agents, because deliberative agent maintains a symbolic representation of the world it inhabits [35], and is capable to plan its actions.

Hybrid agents are defined in [36]. These types of agents incorporate several styles of behaviours, including reactive, deliberative and cooperative behaviours.

An **embodied agent** (also referred to as an interface agent), is an intelligent agent that interacts with the environment through a physical body within that environment; to be considered an 'interface agent', an agent needs to directly communicate with a person through the input and output of the user interface [37], [38]. An interface agent "*accepts user requests, directs them to computer devices or other agents, monitors task execution, and reports back to the person initiating the request. The agent may add graphics or animation to the interface, use speech input and output, or communicate via other sensory devices.*" [39, pp. 1]

Fuzzy agents use fuzzy logic techniques, which have been used for the implementation of several intelligent controllers and for navigation in autonomous vehicles [40].

Mobile agents are agents that are capable of "*transmitting themselves – their program and their state – across a computer network, and recommencing execution at a remote site.*" [5, pp. 193] This type of agents invoke interest of the object-oriented development community, with the basic idea that they would replace remote procedure calls, as a way of communicating between processes over a network.

A group of authors categorizes agents into five classes, based on their degree of perceived intelligence and capability [126]:

- **Simple reflex agents**, which act only on the basis of the current percept, ignoring the rest of the percept history. These are based on the condition-action rule: "if condition then action".
- **Model-based reflex agents** can handle a partially observable environment. They are keeping an internal state that depends on what it has seen before so it holds information on the unobserved aspects of the current state.
- **Goal-based agents**, which are using "goal" information which describes situations that are desirable. This allows the agent a way to choose among multiple possibilities, selecting the one which reaches a goal state.
- **Utility-based agents** measure how desirable a particular state is, which is obtained through the use of a utility function which maps a state to a measure of the utility of the state.
- **Learning agents** are capable of operating in unknown environments and becoming more competent than their initial knowledge alone might allow.

According to [147], an essential ability of an agent must be its ability to learn from experience and hence adapt appropriately. The same authors further argue that this notion implies a system which can adapt and generate its own rules, instead of being restricted to simple automation.

In the context of a self-sustainable human settlement, where there is an assumed implementation of various hardware, software and human interacting elements of a socio-technical system, agents are considered as autonomous entities which are able to perceive certain parameters of their environment and are acting on behalf of their owners, dwelling inhabitants, in order to achieve defined goals.

Socio-technical systems are modeled in the literature by using agent-based modeling in a significant amount of papers. Some of the applications include areas such as cooperation and coordination [129-130], resource allocation [131-132], electronic commerce [133-134], etc. Multiagent Resource Allocation (MARA) is an area of research "*at the interface of Computer Science and Economics*" [233, pp. 1], which requires an interdisciplinary approach. A valuable overview of the area is presented by Chevaleyre et al. [233], introducing four application areas such as sharing of satellite resources, manufacturing control, grid computing, and industrial procurement. Some of the notions from the MARA field are of particular interest for the development of the SSSHS simulation framework; namely, resources (objects that are being distributed among agents), allocations (the specific distribution of resources), agent preferences (for example, an agent A may prefer the resources shared by another agent which has the nearest physical location in regards to the location of the agent A, because of the resource transfer costs perspective; in other example, an agent A may offer its surplus of resources to another agent which has the lowest resource quantities at the moment; etc.), the allocation procedure (the SSSHS framework would facilitate a distributed allocation of resources, in contrast to the centralized allocation), negotiation mechanisms (needed for the complex processes of distributed allocation of resources, etc.

Complex, realistic and large-scale problems are beyond the capabilities of an individual agent [132]. **Bounded rationality** [135] can be applied to individual agents, because their capacity is limited by their knowledge, their computing resources, and their perspective.

The field of distributed/concurrent systems has been investigating properties of systems with multiple interacting components, and there are theories, programming languages and tools developed in order to explain, model and develop such systems. [172-174].

A **multi-agent system** can be defined as a loosely coupled network of problem-solvers (agents) that interact to solve problems that are beyond the individual capabilities of knowledge of each problem solver (agent) [136].

According to [23], a multi-agent system is a system which is composed of two or more agents.

Each agent is individually motivated and attempts to maximize its own utility; the characteristics of multi-agent systems are that (1) each agent has incomplete information, or capabilities for solving the problem, thus each agent has a limited viewpoint; (2) there is no global system control; (3) data is decentralized; and (4) computation is asynchronous [137].

A multi-agent system is one that consists of a number of agents which interact with one another; such a system can be observed as a system with multiple interacting components. In order to successfully interact, these components (agents) must cooperate, coordinate, and sometimes even compete, negotiate and argue with each other.

According to Russell et al. [126], the agent's environment can be classified in regard to its properties as follows:

- **Dynamic and static**. Dynamic environment changes outside the frame of agent's control (for example, the physical world). Static environment is only changed by the actions of an agent; without such actions, it is assumed that it remains unchanged.
- **Continuous and discrete**. An assumption of finite number of actions and percepts identifies the environment as discrete; otherwise, the environment is continuous.
- **Deterministic and non-deterministic**. If there are no uncertainties about the action -> effect relationship, i.e. if any action has a single, well known effect, an environment is considered to be deterministic. Otherwise, the environment is non-deterministic.
- **Accessible and inaccessible**. If an agent can obtain complete and accurate information about the environment, the environment is considered to be accessible.

According to this classification, a self-sustainable settlement in a real world context would describe its environment as dynamic, continuous, non-deterministic and inaccessible.

According to [138], a number of different approaches have emerged as candidates for the study of agent-oriented systems, and one such architecture views the system as a rational agent having certain mental attitudes of **Belief, Desire and Intention**, known by the abbreviation BDI, which represents, respectively, information, motivational and deliberative (decision) components of the

agent. These mental attitudes directly determine the system's behaviour and are critical for achieving adequate performance when deliberation is subject to resource bounds [139], [140].

Beliefs represent the information on the state of the environment; it is a component that should be updated appropriately after each environment-sensing action. This component could be "*implemented as a variable, a database, a set of logical expressions, or some other data structure*" [138, pp. 2]. The main difference between the notion of belief and the knowledge is the assumption that beliefs may be incorrect.

Desires represent goals, or some desired end states. Intentions "*refer both to an agent's commitments to its desires (goals) and its commitment to the plans selected to achieve those goals.*" [81, pp. 9] In contrast to desires, intentions have to be consistent (they can't conflict with each other), while desires may be in conflict with each other.

The BDI model originates in the theory of human practical reasoning developed by the philosopher Michael Bratman [140]. Bratman argues that intentions play a significant and distinct role in practical reasoning and cannot be reduced to beliefs and desires. The conceptual framework of the BDI model is described in the paper [141], and the well-known implementation of the BDI model is the **Procedural Reasoning System** (PRS, Figure 1), developed by a groups of authors [142] [143], which is considered to be the first agent architecture to explicitly implement a BDI paradigm.

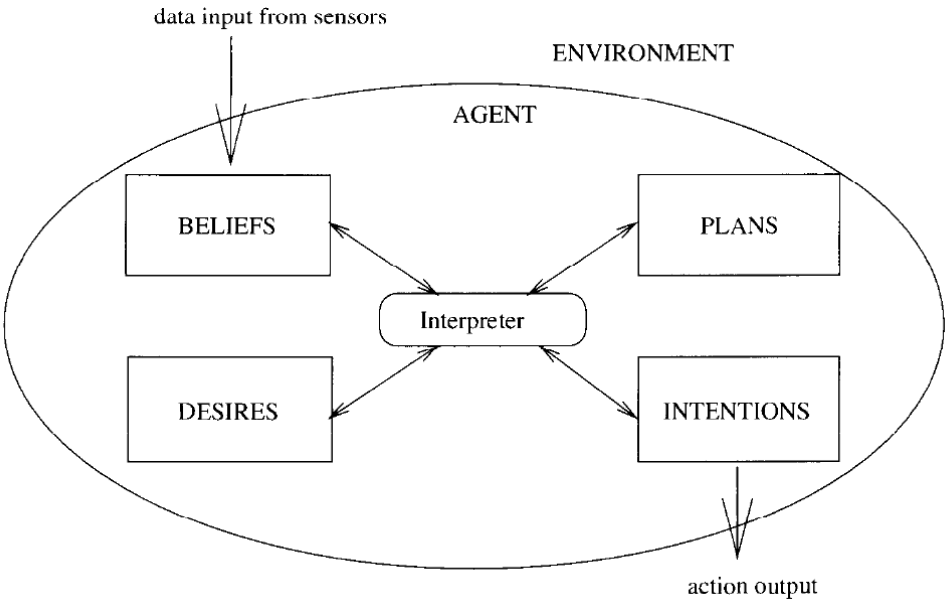

Figure 1. The Procedural Reasoning System [5]

In PRS, an agent is equipped with a library of pre-compiled plans. These plans are manually and preliminary constructed by the agent programmer.

In contrast to multi-agent system approaches (often bound to a computer science perspective on distributed systems), agent based modeling uses similar principles in a more social science oriented perspective.

Agent-based modeling offers an important approach to the simulations of complex socio-technical systems, mainly because of its capacity of representing significant phenomena, usually referred to as "**emergence**", resulting from the characteristics and behaviours of individual agents [144].

Emergence is considered to be a key property of the complex system, although it has "*experienced controversy and confusion*" [145, pp. 3]. It is an idea that intelligent behaviour emerges from the interaction of various simpler behaviours.

A group of authors [146] argues that agent-oriented techniques provide a set of tools and techniques that are particularly suited to "*domains that are functionally or geographically distributed into autonomous subsystems, where the subsystems exist in a dynamic environment, and where the sub-*

systems have to interact more flexibly." [146, pp. 3] They further argue that agent-oriented techniques reduce the complexity in systems design, enhance the robustness and adaptivity of systems, and can be used to structure and appropriately combine the information into a comprehensible form.

Numerous **methodologies** that support the development of agent systems have been proposed. Such agent-oriented methodologies are based on a variety of notations, techniques, concepts, and methodological guidelines, assisting in understanding of and designing a particular system. Some of these methodologies are based on standard methods like UML [82] or CommonKADS [83]. In order to integrate multi-agent systems, UML was extended to AUML [84], and CommonKADS was extended to MAS-CommonKADS [85], [86].

The agent-oriented methodologies could be divided into two groups, based on their roots [5]:

1. Those that originate from the object-oriented paradigm, which either extend existing object-oriented methodologies, or adapt object-oriented methodologies to the purposes of agent-oriented software engineering [87], [88], [89], [90], [91].
2. Those that are based on the knowledge engineering (KE), or other techniques [92], [93], [94]

Figure 2 illustrates the roots of agent-oriented methodologies from the object-oriented methodologies [86]. Figure 3 illustrates a genealogy of proposed methodologies in a more verbose manner, recognizing object-oriented paradigm (OO), knowledge engineering (KE) and Requirements Engineering (RE) as ancestors, which have been extended by agent programming abstractions [95].

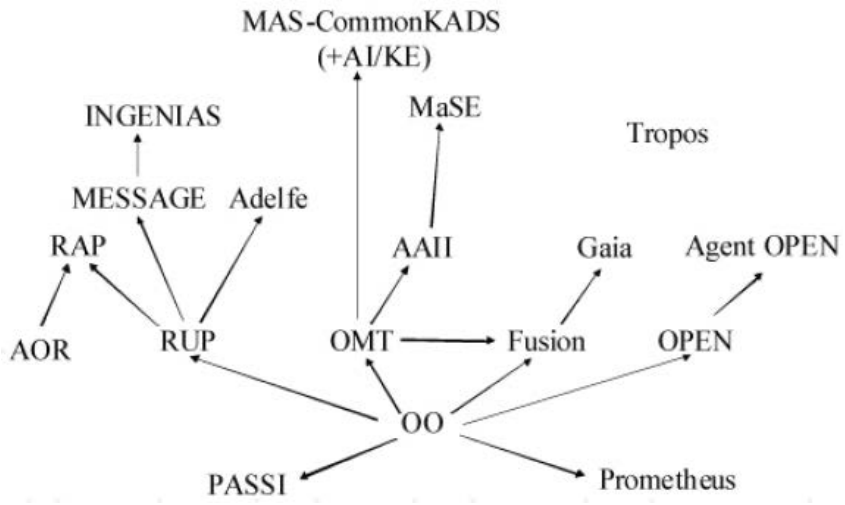

Figure 2. Influences of Object-Oriented Methodologies on Agent-Oriented Methodologies [86]

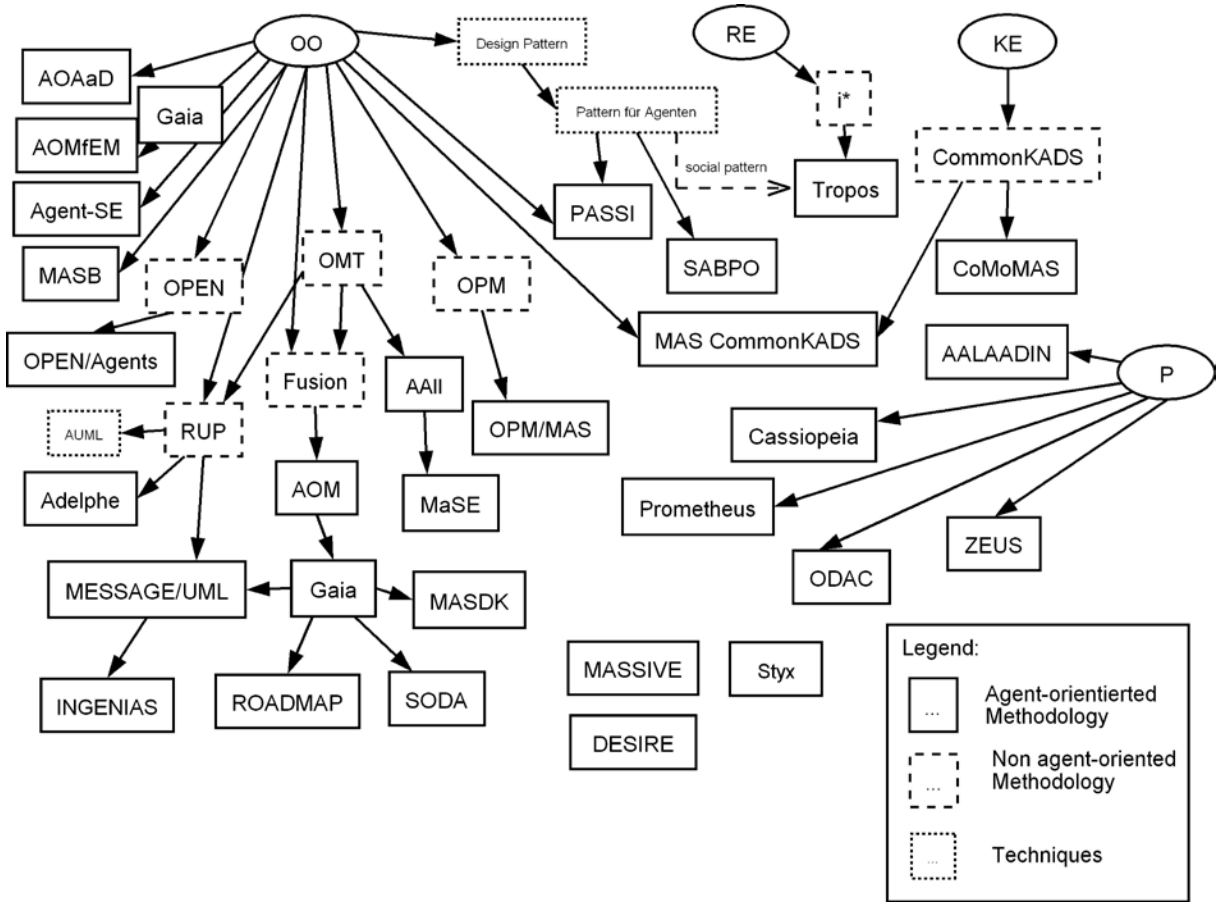

Figure 3. Genealogy of agent-oriented methodologies [95]

A relatively small amount of work has been conducted to compare agent–oriented methodologies, and these approaches are based on two sources of features to examine [95]. In one of such sources, general software–engineering criteria is adopted. In other, specific criteria that are needed to support agent–oriented concepts in development were identified. Figure 4 for example illustrates a comparison of chosen agent-oriented methodologies by their organizational features [96].

Organizational features		*Roadmap*	*AGR*	*Civil Agent Societies*	*E-Institutions*	*SODA*	*MESSAGE*	*INGENIAS*	*GaiaExOA*	*Tropos*	*OperA*
Objectives		√				√	√	√	√	√	√
Structure	Topology								√	√	√
	Roles	√	√	√	√	√	√	√	√	√	√
	Interactions	√	√	√	√	√	√	√	√	√	√
	Social Norms		√	√	√	√			√		√
Dynamics	Agent Joining		√	√	√	√					√
	Role enactment	√	√	√	√	√	√	√	√	√	√
	Behaviour control		√	√	√			√	√		√
Environment		√				√	√	√	√	√	
Implementation					√			√		√	

Figure 4. Comparing agent-oriented methodologies via selected organizational features [96]

The development of the SSSHS framework will use an adapted workflow described in Multiagent Systems Engineering (MaSE) Methodology [97], because it offers a blueprint for a step-by-step construction of a multiagent system through an entire software development lifecycle, which starts from the problem description and specification, and ends in a concrete implementation of an agent-based system. MaSE methodology is presented in Figure 5.

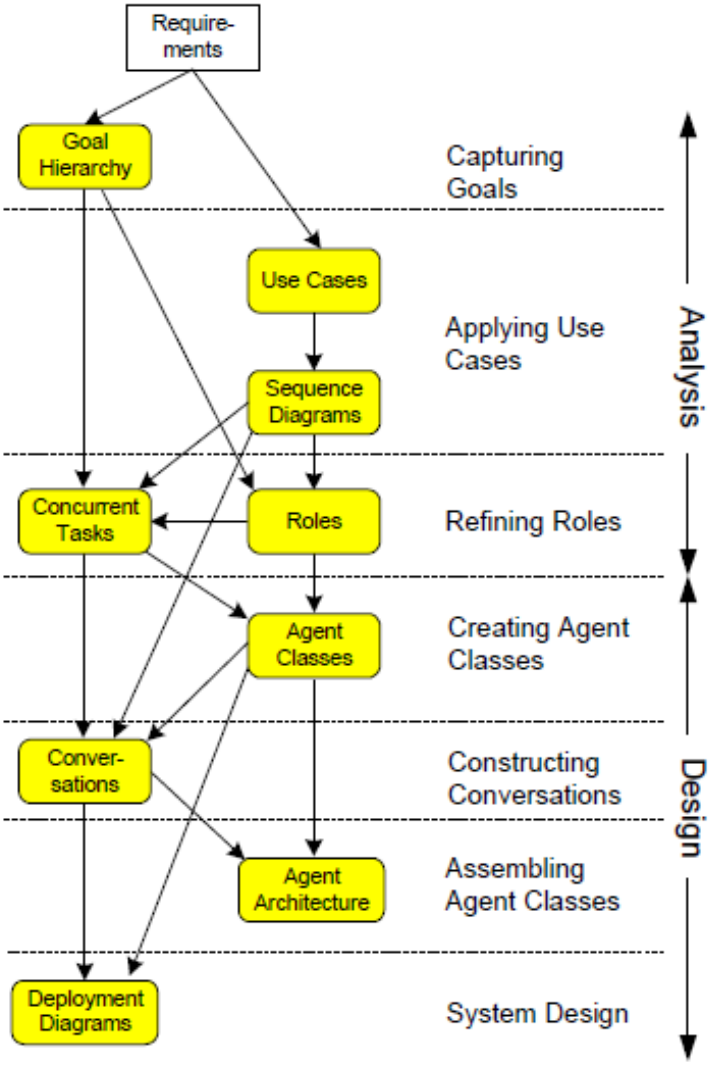

Figure 5. The MaSE Methodology [97]

3.5.1 Cooperation and Conflicts among Agents

Agents in the multi-agent system may be designed by different individuals or organizations, thus representing their interests; those agents are self-interested and it cannot be assumed that they share a common goal. Because of this, one agent's interests may **conflict** with those of others [5]. Research is therefore concerned with questions such as why and how agents cooperate [41] in order to achieve their goals, how does a group of agents work together to solve problems [42], how can agents recognize and resolve conflicts [43], [44], [45], or how can agents negotiate [46], [47].

According to [48], three phases may be identified during the cooperative problem solving:

1. Negotiation phase (tasks are allocated to other agents)
2. Execution phase (problem is being solved)
3. Results-reporting phase (report if the task was successfully solved, or not).

In a network of communicating problem solvers, the **Contract-Net protocol** is a high level protocol for achieving efficient cooperation through task sharing [49] [50] [42]. The basic principle of this protocol is contracting, with a set of components that can assume tasks dynamically according to the contract. The agent that generates the task advertises its existence to other agents, which consider it, and reply with their conditions upon which they can execute the task. The task advertising agent chooses the best offer.

The SSSHS framework is designed with the **benevolence assumption** on cooperative problem solving [5] – the agents in the system implicitly share the common goal, and there is no potential for conflict between them. The agents are designed so as to help out whenever needed, and the overall system objective prioritizes their cooperative behaviour.

Game theory is a mathematical theory involved in a research of interactions between self-interested agents [175], and it plays a major part in use for the analysis of multi-agent systems. Wooldrige argues that many of the solution concepts developed in game theory "*were developed as descriptive concepts, without a view to computation*" [5, pp. 14]. Multi-agent systems research allow the use of tools of computer science such as computational complexity theory. [176-177]. However, game theory become to be predominant theoretical tool in use for the analysis of multi-agent systems, particularly in researching problems regarding negotiations [5].

Faratin et al. [51] present a formal model of negotiation between autonomous agents. Negotiation is defined by Pruitt [52] as a process by which a joint decision is made by two or more parties. The parties first verbalize contradictory demands and then move towards agreement by a process of

concession making or search for new alternatives. In a service-oriented negotiation, an agent (the client) requires a service to be performed from some other agent (the server). In the SSSHS framework, triggering the lower threshold value of the resources in a dwelling unit would initiate a series of mechanisms to be performed in order to prevent the complete loss of resources, with the last mechanism in the form of a negotiation processes with other agents, where an agent would require a service (a resource transfer) from other agents. Some of the more relevant characteristics of the SSSHS negotiation processes would be the following:

- Individual agents could be both clients and servers in different simulation contexts.
- A service could be provided by more than one servers.
- The client has the weaker negotiation position by default, because it is assumed that the defined values of the individual comfort levels of the servers are the threshold values of the negotiation offers.
- The initial weaker position of the client is alleviated by the possibility of the client to repeat its modified request to more than one server.

The client would send requests sequentially to the servers according to the lowest-loss rule – initial request would be sent to the server from whom the resource loss in the transfer process would be the lowest. This could assume for example the shortest physical distance between the two agents, or the quality of the infrastructure between them.

According to [53, pp. 1], „*Conflict resolution is one of the most important aspects in distributed systems and multi-agent systems.*“

Within a multi-agent system, conflicts are usually seen as disturbances [128]. The definition of a conflict is presented in [54, pp. 30], and according to [128]: „*A conflict is a subset of all propositional attitude(s) (e.g. beliefs, desires, intentions, hopes, etc.) of the agent that must be reduced by removing a propositional attitude*“, which refers to the situation with exclusive, or incompatible, attitudes. According to Aïmeur [55], the three possibilities for conflict resolution are the following:

1. **Negotiation**, where conflicted parties are trying to reach a common agreement through a discussion process.
2. **Mediation**, where a neutral party is facilitating the research for possible solutions.
3. **Arbitration**, where a neutral, unbiased party decides on a solution.

"*Thereby conflict resolution is the application of certain techniques for the transition from a situation with conflicting agent attitudes to a situation with less or no conflicting agent attitudes*." [54, pp. 31].

Huhn et al. are arguing that conflicts are "*perceived as unpleasant and lead to an imbalance in the environment*", but also, that conflicts are "*decisive for the evolution (among other things, they cause long-term changes) of humans, teams and organizations and must be accepted as part of the development (evolution) of, e.g., a system or a society*.", and that "*conflict avoidance is not always the best choice*" [56, pp. 19].

Tessier et al. presents a **Conflict handling action model** (Figure 6) and argue that the model is "*essential in that it comprises the basic entry steps for conflict handling, it respects the different strategies to deal with conflicts, it uses elementary action points to vitalize the strategies, and it ends up with a learning step*" [128, pp. 26-27]. The model presents the standard conflict handling activities from conflict recognition to the conflict process analysis, in respect to the conflict evolution process.

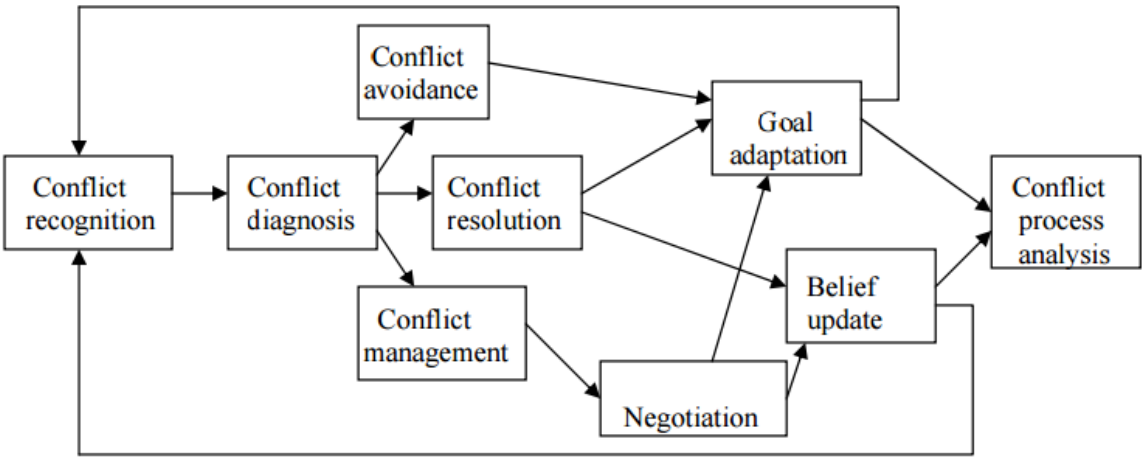

Figure 6. Conflict handling action model [128]

Tessier et al. [128] have published a collection of paper that are researching various aspects of conflicts in agent systems like conflicts between collaborative agents [57], [58], definition and categorization of conflicts [59], and conflicts in the context of sociology [60], [61].

The research paper [59] categorizes conflicts as physical conflicts and knowledge conflicts. **Physical conflicts** refer to resource conflicts, where agents' interests are affected by insufficient, not simultaneously shareable, resources. There had been research conducted on effective resource sharing in several papers such as [62], [63] and [64].

Knowledge conflicts refer to epistemic conflicts where agents have different interpretations of the environment and their own desires [65]. It is a situation where "*several agents have their own local knowledge base with incomplete and different information about the same domain. Moreover, each agent is responsible for maintaining the coherence of its knowledge base. Then, whenever agents exchange information, inconsistencies may arise in their local knowledge bases due to these different and/or incomplete agents' views of the state of affairs.*" [66, pp. 1].

Merging potentially conflicting beliefs between agents is presented as a way to resolve knowledge conflicts [67], [68], [69], [70].

Also, research on **argumentation** has demonstrated the use of argumentation frameworks for conflict resolution [71], [72], [73], [74].

Classifications of knowledge conflicts divide those conflicts into two general categories [75]:

1. **Syntactic** conflicts (contradictions) – the fact and its negation;
2. **Semantic** conflicts – "*stick to the considered domain (e.g., classifying an object as square and rectangular in a domain where objects are restricted to be classified by a unique shape)*" [66, pp. 1]

Research on solving knowledge conflicts is being conducted both in syntactic conflicts area [76], [77], [78], and on semantic conflicts [79], [80].

In the paper [66], authors have proposed an approach for enabling agents to detect and solve both types of conflicts indistinctly, by a "*combination of a formal knowledge model represented through ontologies and a mechanism for managing conflicts based on argumentation techniques*" [66, pp. 1].

3.6 Internet of Things

The Internet of Thing (IoT) is a novel field defined as "*a variety of things or objects (...) which, through unique addressing schemes, are able to interact with each other and cooperate with their neighbors to reach common goals*" [178, pp. 1].

These "things" include numerous different physical devices including but not limited to various kinds of sensors and actuators, mobile devices, TV sets, car/vehicle computers; but also non-ICT appliances (dishwashers, microwave ovens, refrigerators), electrical energy sources and building components [179].

An agent-based modeling approach shows to be adequate in the modeling of Internet of Things [149], [180], especially considering the possible interoperability issues coming from heterogeneous set of devices, different communication protocols and underlying networks.

Some of the key application areas of IoT are smart cities [181], smart power grids [186], smart health [187], smart transport [188], as well as smart buildings [189] which includes smart living solutions [190].

Publications covering these areas were considered in dealing with smart self-sustainable communities, since they cover detailed theoretical and practical models of implementing smart solutions for residential facilities, as will be described in the following section.

3.7 Intelligent Buildings and Smart Cities

Authors in [147] define an intelligent building as one that uses computer technology to autonomously operate the building environment for optimization of energy consumption, user comfort, safety and monitoring functions.

Intelligent building/houses/homes involve computer technology (for example, multi-agent systems) which regulates building components, utilities, electrical circuits, and heating, ventilating

and air-conditions systems in order to monitor building functions, security, energy consumption, and provide a comfortable environment to users [148].

[149] further argues that intelligent buildings should not only be modeled as multi-agent systems (MAS), but should include learning capabilities to adapt to the user's need and changing preferences.

Watson et al. in [150] propose the development of a new subfield of IS called energy informatics, which recognizes the role that information technologies could play in reducing energy consumption (resulting as well in reducing CO_2 emissions) and they present their core idea with the following illustrative expression:

$$Energy + Information < Energy \quad (1)$$

Rutishauser and Schafer [151] argue that the environment of intelligent buildings is very complex: inaccessible, non-deterministic, non-episodic, dynamic and continuous, and it is further suggested that a feasible solution for controlling it would be to implement adequate multi-agent systems [152].

Similarly, it is suggested that if a problem domain is "*particularly large, complex, or unpredictable, then the only way it can reasonably be addressed is to develop a number of functionally specific and (nearly) modular components (agents) that are specialized at solving a particular problem aspect.*" [153].

Relating to the above mentioned modular components, it would prove beneficial for a modeled self-sustainable system to have the possibility to be open-ended and extensible, as there might be a need for adding new resource sources, or removing existing loads at any time, without having an impact on the overall operation of the system. In real world scenarios, such dynamics is not only possible, but very probable. For example, new modular photovoltaic panels can be added on the roof of the house, adding new electrical capacity to the dwelling. Some residents of the house could move out for a prolonged period of time, or permanently, thus removing a part of the resource

consumption load. Water reservoirs, or electric battery packs could be added if existing storage solutions prove to be inefficient for the needs of the dwelling. A proposed framework should be able to provide such flexibility in the system design modelling before each simulation run.

In [147], authors have defined a set of behaviours for an agent responsible for autonomously governing the building environment:

- *Safety* behaviour (for example, light levels are always at safe levels).
- *Efficiency* behaviour (for example, lights should be turned off when there is no relevant presence in the room, or if there is sufficient natural light present in the room).
- *Emergency* behaviour (controls resources in an emergency situations).
- *Manual* behaviour (complete control of the user; the system reacts to any orders the occupant gives it).
- *Self-taught* behaviour (system adapts to different users and environments).

The agents in the mentioned paper do not contain complex modelling or reasoning capabilities, because these might be very processor-intensive. Whilst limited, this approach ensures that the system is able to respond in real time on any situation or event.

Similarly, a group of authors [154] discuss agent behavior's decomposition into 4 behaviour subsets, and define hierarchy amongst them: "safety" behaviour has the top priority, following by "emergency", "efficiency", and "comfort" behaviours. However, their architecture is not limited to these pre-defined behaviours, but it also "*has the ability to learn new behaviours dynamically, based on actions taken by occupants within the room*." [154, pp. 4] Learning mechanism is derived from the CASE based learning, a branch of traditional AI work [155]. In designing the SSSHS framework, "safety" behaviour could be implemented as a pre-defined variable, not admissible to change by the simulation user, or as a lower threshold in the usable variable range. It is a constant, and takes priority above all other behaviours, which depend on the current situation parameters. "Emergency" behaviour would, for example, initialize if the resource levels are dropping near the predefined lower threshold values. It is an exceptional behaviour, triggered by a certain event in a

simulation run. If no such event would take place, a “common” behaviours, “efficiency” and “comfort” are run by the default, but with constant respect to the “safety” parameters.

MavHome (Managing an Intelligent Versatile Home) is a project supported by National Science Foundation which is focused on creating a home that acts as an intelligent agent, and is described by a group of authors in [156]. Authors introduce the *MavHome* architecture, discuss the role of prediction algorithms, and present a meta-predictor which combines the strengths of multiple approaches to inhabitant action prediction. Authors define the need for a number of capabilities for their smart home scenario and its integration: machine learning, mobile computing, robotics, databases and multimedia computing.

Blumendorf [122] raises the question *Can we make smart home sustainable or sustainable homes smart?* and discusses current trends and challenges arising with these questions. Along with illustrative examples of real-life efforts of designing sustainable homes, author elaborates critiques and problems regarding smart homes, like increased energy consumption, electronic waste (*e-waste,* discussed in [157], user frustration and cognitive overload [158], and similar, which need to be addressed in order to make smart home sustainable.

Georgakarakou [152] found 72 literature sources identified to have potential significance on some aspect of agent technology in intelligent buildings domain.

Smart cities are one of the important application areas of Internet of Things [181], and their main focus is on “*applying the next-generation information technology to all walks of life, embedding sensors and equipment to hospitals, power grids, railways, bridges, tunnels, roads, buildings, water systems, dams, oil and gas pipelines and other objects in every corner of the world, and forming the “Internet of Things” via the Internet.*” [182, pp. 1].

The term “smart city” (referred to also as the “intelligent city”) is not formally defined, and has various meanings. Komninos [183] has identified four different descriptions of the term “intelligent city” through the literature, relating to (1) the virtual reconstructions of cities; (2) “smart growth” and “smart community” - a development based on information and communication technologies;

(3) environments with embedded information and communication technologies, such as embedded physical sensor systems, which would facilitate computation in an everyday physical world; (4) "*territories that bring innovation systems and ICTs within the same locality"* [183, pp. 13-20], with talented individuals, institutions and digital innovation spaces as main buildings blocks of such environment.

A smart city is designed to make intelligent responses to various kinds of needs in the context of daily livelihood, environmental protection, public safety, city services, etc. [184]

According to the visionary approach presented in [185], "*Cities are becoming smart not only in terms of the way we can automate routine functions serving individual persons, buildings, traffic systems but in ways that enable us to monitor, understand, analyze and plan the city to improve the efficiency, equity and quality of life for its citizens in real time*". [185, pp. 2]

In the context of utilization of sustainable resources in remote rural areas and decentralized energy systems, authors [234] are working on the improvement of the transport efficiency of sustainable resources by focusing on the nature-inspired behaviors of autonomous agents, presenting a self-organizing mechanism, which can achieve an optimal transport efficiency by tuning a control parameter. This work might prove useful in planning a transportation network in self-sustainable settlements, from a perspective of self-organizing systems.

La Rocca explores an agent approach in integrating the energy system and the physical space structure. The author aims to "*define a relationship between the design of the energy system and the suitable organization of the physical spaces*." [235, pp. 1]

Sen defines an agent community as a "*stable, adaptive group of self-interested agents that share common resources and must coordinate their efforts to effectively develop, utilize and nurture group resources and organization.*" [236, pp. 1]. The work presents a summary of some of the research results on trust-based computing, learning, and negotiation that will "*enable intelligent agents to develop and sustain robust, adaptive, and successful agent communities*." [236, pp.1]

Polaków and Metzger [237] describe the agent-based approach to the operating control task for the wastewater treatment process in the context of biotechnological processes; authors describe the implemented control system, based on a real-time agent communication protocol, utilizing a

blackboard knowledge system. A MAS-based approach supports dynamic reconfiguration, resulting in a better flexibility and modularity of the system. A particularly useful insight for the development of self-sustainable settlements might involve LabVIEW environment and the agents accessing the hardware with the OPC interface.

3.8 Eco Feedback

It has been researched that supplying consumers with information about their energy usage can lead to changes in usage patterns, and decrease in overall consumption. [159-160]

A group of authors [161] claims that the use of real-time feedback can decrease energy consumption by 10% - 20%.

Froehlich, Findlater and Landay [162] have identified 133 papers (89 papers from environmental psychology and 44 papers from the human computer interaction and ubiquitous computing literature) that are reporting about eco-feedback. It is argued that the monitoring of energy consumption has the potential to make users aware of the hidden details of their current behaviour, as well as about how the resident compares to other community members.

When receiving eco-feedback data, users can clearly identify their consumption footprint, and they subsequently become more able to control their energy usage. At the same time, eco-feedback is informing the individuals about resource usage of other residents, which enables an increased social awareness in the household. [163-165]

Yun [165] presents a study where a minimal in-home energy consumption display encourages users to identify high-power devices in their home, and to reduce energy consumption.

A group of authors [166] designed, implemented and evaluated an interactive visualization that allowed users to engage with and understand their consumption data. Validation of this design included 12 participants who installed meters in their homes and used the system for a period of two weeks. The research results showed how the users started to relate energy consumption to

activities, rather than just appliances, and were able to discover that some appliances consume more than they expected by their own personal estimations. The results also indicated that there is a "*potential for interactive eco-feedback technology that engages users with their data beyond mere presentation*" [166, pp. 9].

3.9 Load Management

Load management mechanisms, and its possible connection to multi-agent systems, are presented in [167], and according to distribution control described in [168]. These mechanisms allow time adjustments of certain services/appliances in order for them to operate in time periods when global resource consumption is lower, and/or resource price is lower.

To achieve this, a load management system designer must identify services/appliances that could delay or advance their operation time(s), ones that could have their consumption rates reduced, ones that could be programmed to run according to the weather forecast, etc.

Demand side management is a method designed to coordinate the activities of energy consumers and energy providers. This method seeks to avoid so called peaks of energy consumption [168].

This method could be valuable to the modeling of self-sustainable systems, because it could prevent consumption peak times which have the capacity to deny the system of the self-sustainability property (extremely high resource demand in the short time interval).

The main idea is to smooth the energy consumption by using intelligent domestic appliances that communicate and coordinate with each other.

As an illustrative example, software agents could be embedded in the appliances, and they could negotiate about which of them needs to be temporarily interrupted in the case of unscheduled events like resource shortage, or unforeseen resource consumption (this is called "intelligent load shedding" [98]). Each of the agents can permanently monitor satisfaction levels of its services, and take appropriate actions according those levels.

In the SSSHS framework context, each of the consumption units/agents could have predefined relative priorities. According to these priorities, and if current circumstances trigger such events, consumption units could be asked one by one to reduce, or delay, their resource consumption, thus activating a load shedding mechanism in the SSSHS framework.

The same paper presents a system for power consumption distribution in private homes, which distributes power uniformly over time. In such a system, logical relation is established between devices, and energy consumption policy is coordinated between them. These coordinations and actions are taking place in a distributed, autonomous infrastructure.

Figure *7* illustrates the short-term power management of appliances, where energy consumption peak is avoided [168]:

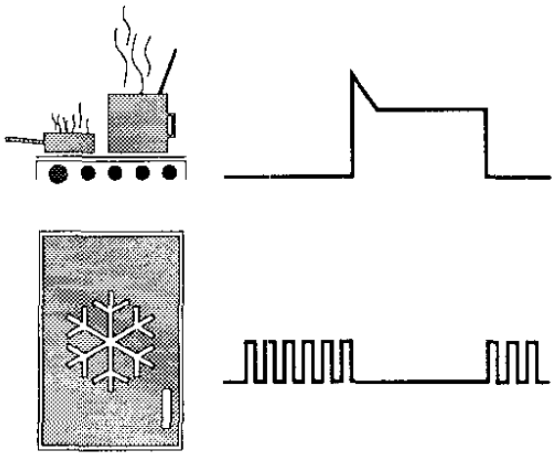

Figure 7. Short-term power management of appliances for avoiding peak consumption - illustrative example [84]

Shift of energy usage load for avoiding peak times in order to reduce the need for high production capacities has been identified as one of the key factors for energy efficiency according to the group of authors [169].

Similarly, objective of the study in [167] was to "*design a building electric energy management system able to determine the best energy assignment plan, according to given criteria*". [167, pp. 2]

Based on a large literature overview and documented cases, a group of authors [170] concludes that the "*main operational benefits of installing intelligent building components include:*

(i) *energy efficiency and higher environmental sustainability;*
(ii) *increased user comfort and productivity;*
(iii) *improved safety and reliability;*
(iv) *improved operational effectiveness; and*
(v) *enhanced cost effectiveness.*" [170, pp. 9]

Authors also suggest that the input that environment gives to agents can't be specified in advance. They gave an illustrative example: "*something happens to one system (e.g. reducing light level) may cause a person to change behaviour (e.g. sit down) which in turn may result in them effecting another systems (e.g. needing more heat); and people are essentially non-deterministic.*" [170, pp. 2] Since there is a myriad of home appliances which consume resources, they have proposed grouping all these operations of different appliances into the term "service". This "service" transforms energy in order to meet a user's need via one or several appliances, and could be qualified as permanent, or temporary, depending on its usage of energy.

Important notion from the same paper is the "satisfaction function", which defines "user comfort" in terms of delivered service regarding the user's feelings, which could be closely related with the notion of personal satisfaction from [171].

4. Framework Implementation

The development of the SSSHS framework uses an adapted workflow described in the Multiagent Systems Engineering (MaSE) methodology [97], which is presented in Figure 5. Outputs of each section in the MaSE methodology are used as inputs for the next section; for example, identified goals are used for the creation of roles; roles are used for the definition of classes; etc. The methodology enables a linear progression of the system development, but is iterative across all phases. Elements of the MaSE methodology such as task state or deployment diagrams were purposefully omissed because of their specifities regarding a concrete model rather than a higher level of abstraction required by the development of the SSSHS framework.

The first phase of MaSE methodology identifies goals of the system in the section 4.5, which are derived from the global system specification, and system specification construction was partially aided by the informal interviews conducted with self-sustainability experts, and by the on-site analysis of the physical system. In the second phase (section 4.6) agent roles are described, and according to those roles, the third phase defines agent classes (section 4.7).

4.1 Consulted Self-Sustainability Experts

In pursue of achieving more realistic context in the process of designing the system, as well as in designing the test-bed scenarios, three individuals, which are considered to be experts in self-sustainability, were informally interviewed in order to gain more insight into the problem domain and to gather relevant simulation data. These interviews were unstructured, i.e. most of the questions were not prearranged and were developed during numerous conversations on the topic of self-sustainability. Almost all of the interviews occurred via email; an opening dialogue with the Consultant #3 took place in the self-sustainable eco-village founded by the consultant.

While most of the questions were developed spontaneously, all the interviewees were asked general, basic set of questions:

1) "*Do you think that a self-sustainable human settlements need a smart resource management as described in the nutshell of this research*"?

2) "*What do you think about self-sustainable mechanisms used by the framework in this research; do you argue that there is anything that the research is critically missing*?"

3) "*What self-sustainable mechanisms do you use in your self-sustainable settlement/dwelling?*"

4) "*Your thoughts and suggestions on the Scenario #1: Permaculture / self-sustainable eco-village*?"

5) "*Your thoughts and suggestions on the Scenario #2: Space colony*?"

6) "*Your thoughts and suggestions on the Scenario #3: Prolonged space flight*?"

Interviewees were also asked about their specific technologies and solutions implemented and used in a daily life. The described technologies and solutions were revised and considered in the SSSHS design process, as described in later chapters. The short biographies of three consultants are available in Appendix A.

4.2 Framework Visual and Narrative Description

System visual description illustrates basic building blocks of the framework. The architecture of the proposed system is composed of two layers; on the top layer, a smart self-sustainable system is composed of individual dwelling units, interconnected in a network infrastructure which allows both data and resource exchange (Figure 8).

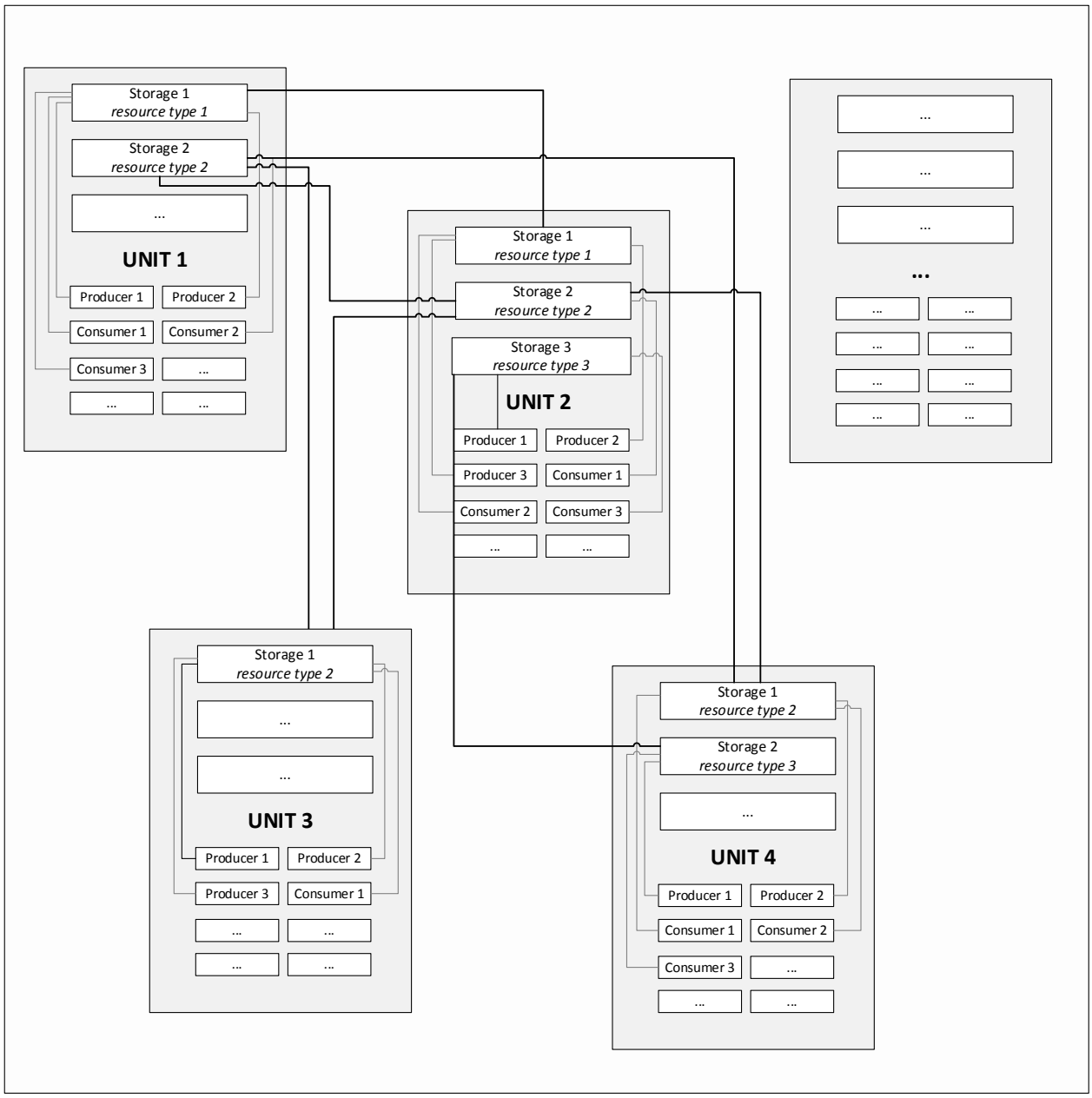

Figure 8. Illustration of the SSSHS framework basic building blocks

Individual dwelling units initiate a communication and resource transfer negotiation only upon the resource depletion within their own subsystems. Proposed network topology of this model is an ideal fully connected network topology. Since this topology ensures that all the dwelling units are connected to each other both by data and resource infrastructure networks, it is possible to connect

and transfer resources from any and to every other dwelling in the settlement system, between storages that process the same resource type.

In the lower layer (Figure 9), each dwelling unit is composed of **agent types**, which include a producer's class, a consumer's class, and a storage's class. As such, individual dwelling units are able to produce, store and consume resources independently.

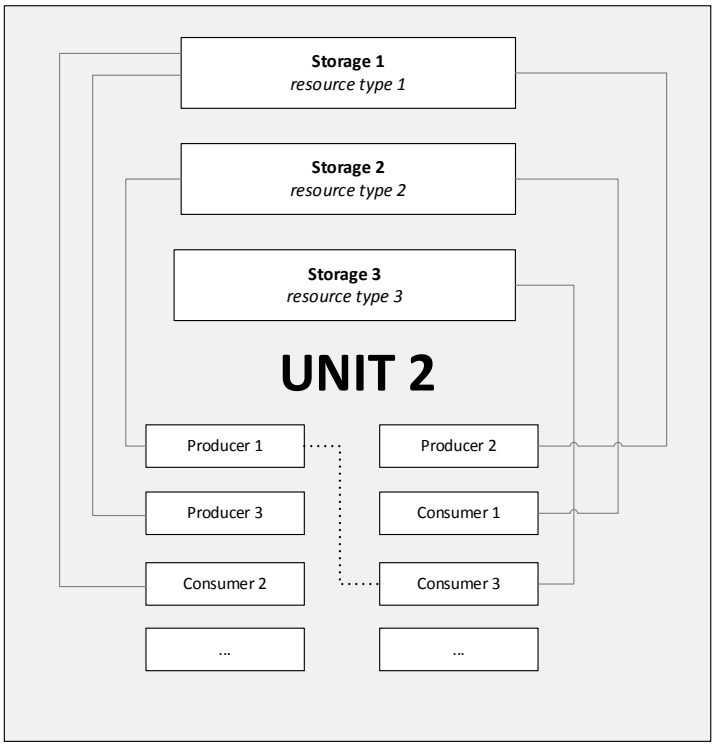

Figure 9. Illustration of a single dwelling unit (detail)

Each agent class has its own parameters/attributes and methods/functions which have the ability to change the inner state of the agent, i.e. to change certain parameter values of the instantiated agent class. Individual dwelling units can implement one or more storage units, each dealing with different resource types. This would facilitate the ability of the SSSHS framework to simulate more than one resource type at a simulation run, describing for example scenarios where the production of one resource type requires the consumption of other resource type (or more than one resource

types). As presented in Figure 8, communications and resource transfers between individual dwelling units is possible through storage units which process the same resource type.

4.3 Self-Sustainability Mechanisms

Self-sustainability mechanisms in the context of this research can be described as those methods, selected and activated by the framework, which have the ability to prolong the self-sustainability property of the modelled system. The set of these methods is derived from several sources:

- Reviewed literature on the subjects including but not limited to intelligent buildings, eco feedback, load management, resource consumption, self-sustainable human settlements (presented in chapter 3);
- Informal interviews with three experts on self-sustainability (presented in section 4.1.);
- Real-world resource distribution mechanisms used by the gas and electric companies (such as [197]);
- Knowledge of the operation of a number of widely used consumption units, personal experiences and analysis.

Based on their triggering events, self-sustainability mechanisms fall into two possible categories: *lower threshold mechanisms* (triggered upon the event of shortage of resource), and *upper threshold mechanisms* (triggered upon the event of overflow of produced resources). Both categories involve a series of actions to be executed in order to “stabilize” the system, i.e. to maintain the self-sustainability.

4.3.1 Lower Threshold Mechanisms

Lower threshold mechanisms are those self-sustainability mechanisms which are triggered upon the event of resource level reaching a predefined lower threshold value. The lower threshold value

is set by the user when designing the system, and can be manipulated with subsequently in the efforts to achieve needed simulation results.

When resource level reach the lower threshold value, the framework would alert to the possibility of losing the resources completely, consequently labeling the system as non-self-sustainable and exiting the simulation. Following the alert, which describes the event in more detail (where exactly did the event occur, what is the current level of resources, the value of lower threshold) the framework would trigger the set of actions directly aimed towards the global goal of the system, i.e. to retain the self-sustainability property. These mechanisms are described hereafter.

4.3.1.1 The “Economy mode”

In the proposed resource management simulation framework it is assumed that consumer units have the **ability to reduce their consumption rates** below their default consumption values. A system modeler defines this ability upon instantiating agent classes. The example of this mechanism is used by PG&E Corporation in the following message to their consumers: “*If there is an energy supply emergency, between May 1 and October 31, PG&E will send a signal to activate your SmartAC device, which allows your air conditioner to run at lower capacity to help avoid power interruptions.*" [197]. SmartAC is a proprietary device which is connected to the customer’s air conditioner, and it can be remotely activated with a goal to manipulate an air conditioner’s compressor cycle, turning it on and off in short increments of 15 minutes of every half hour. This mechanism prevents the California’s electrical system to be used beyond its capacity, possibly causing critical power interruptions.

According to FERC (Federal Energy Regulatory Commission, United States federal agency), demand response (DR) is defined as “*Changes in electric usage by end-use customers from their normal consumption patterns in response to changes in the price of electricity over time, or to incentive payments designed to induce lower electricity use at times of high wholesale market prices or when system reliability is jeopardized.*” [198]

Demand response includes several more feasible possibilities of interventions than merely altering the consumption rate: “*demand response includes all intentional modifications to consumption*

patterns of electricity of end-use customers that are intended to alter the timing, level of instantaneous demand, or the total electricity consumption“ [199]. Some of these factors, namely altering the timing, are implemented as an integral part of the SSSHS framework, as will be discussed in forthcoming chapters.

The proposed abilities for temporary reduction of the consumer’s consumption rates are based on the individual real-world consumer units and resource consumption policies. For example, many household appliances such as washing machines, dishwashers, ovens, etc., have several different programs available, and one of them is often able to operate with reduced consumption of resources such as electricity, water and similar - and often called the “**economy**” program. Because of the useful association with the real world scenarios, a term “economy mode” will be used in further framework description.

It is assumed that default consumption rates set by the modeler are initialized according to modeler’s real-world system’s **targeted comfort levels** (i.e. ambient temperature, daily water usage per capita, etc.), and/or physical unit’s characteristics, available modes of operation and the resulting efficiency of each mode. Setting the possibility for reduced consumption rates in regard to the default consumption rates, can refer to the possibility for reducing the ideal comfort levels, or reducing the unit’s efficiency regarding the end result, depending on the modelled scenario at hand, and it is up to the modeler to quantify allowed reduced consumption rates for each consumer unit.

A household could define specific internal **resource consumption policies**. For example, if the default agreed upon value for ambient temperature is 22°C, members of the household could agree on “economy” ambient temperature that, in the time of a resource shortage, can be decreased for a certain value in regard to the default value, for example for 2°C, resulting in a new ambient temperature of 20°C. Showering resource expenses can be reduced in terms of time, lower water pressure, or faucet aerators, resulting in a reduction of water consumption. Ambient illumination levels could be reduced by dimmers if possible, optimizing illumination levels according to specific rooms’ defined purposes and residents’ presence. There is a considerable amount of similar factors

to be taken into consideration for a modeler when modelling specific scenario in the SSSHS framework.

4.3.1.2 Manipulation of operating times: delaying

Similarly to the “economy mode”, modern appliances often have the programmable ability to **postpone** their starting times, usually defined in hours. A machine can be manually delayed, or started **earlier** than scheduled, or its operating times can be controlled by an embedded agent unit. Some modern appliances have the ability to directly input their starting and stopping operating time. Delaying could also be described in the context of aforementioned demand response mechanisms.

Resource consumption policies established by residents inside the dwelling units can specify behaviours of residents regarding the use of resources, for example specific resident’s activities that consume resources could be postponed to time periods when resource levels are beyond the lower threshold zone.

4.3.1.3 Resource Negotiation with Neighboring Sub-Systems

The previously described mechanisms are confined to their sub-systems (individual dwelling units). **If these fail to maintain the self-sustainability of their sub-system, an inter-dwellings mechanisms is triggered by the framework, called “resource negotiation”**. This mechanism calculates the value of needed resources which would bring the sub-system’s resource level above the lower threshold zone, reducing the risk of the system to become non self-sustainable. According to the service-oriented negotiation paradigm, an agent (the client) requires a service to be performed (a resource transfer) from other agents (servers). The client presents its request to the server according to the calculated value of critically needed resources, and in respect to the server’s own resource levels. The server then calculates its own capacities based on the most current parameter values stored in its inner state, and decides on the counter-offer it is willing to give to the client. If the counter-offer is greater than zero and resource transfer costs are acceptable, the negotiation thread is initiated. If the resource transfer costs are not acceptable, or the agreement is achieved but the counter-offer does not provide sufficient resource quantity, the client sends further requests

to other agents, sequentially, until its resource levels are above the defined threshold. The client negotiation perspective is depicted in Figure 10 via **finite state machine**.

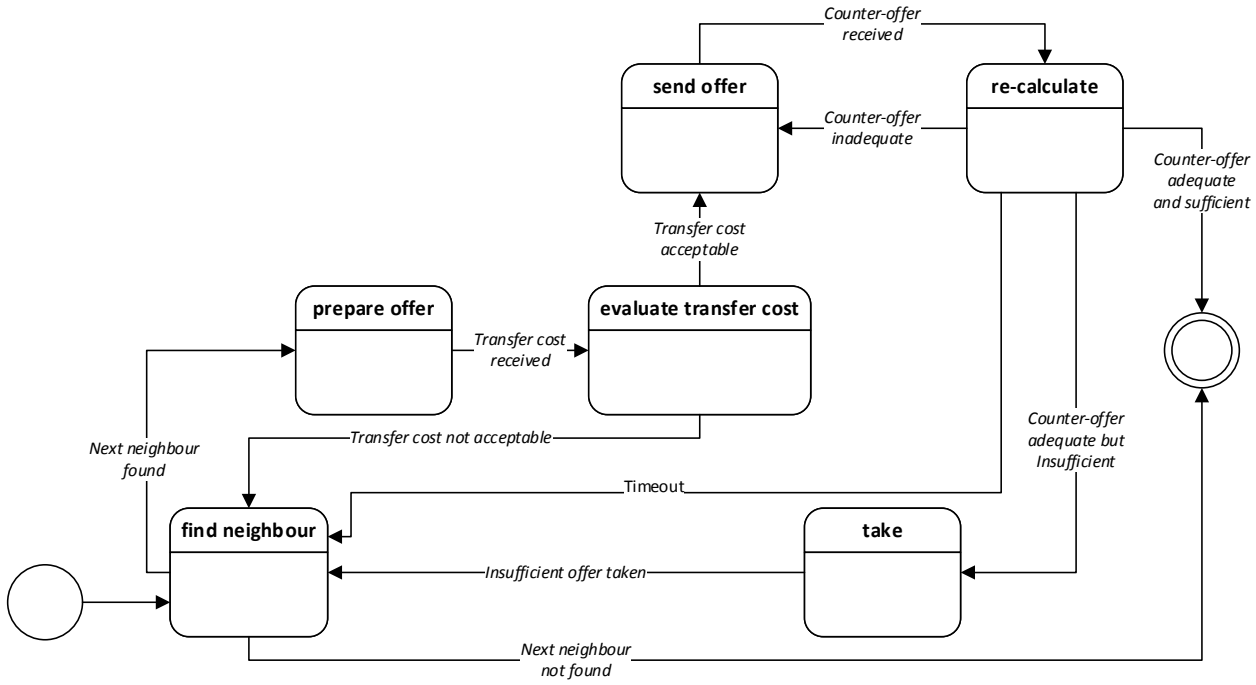

Figure 10. A finite state machine of the negotiating client

This “resource negotiation” mechanism assumes that the complete data and resource infrastructure network between dwelling units in the system has been established, and that individual dwelling units can both communicate and transfer physical resources via this network. As such, this is the most hypothetical and most demanding assumption regarding the real world scenarios and implementations in existing human settlements on Earth.

According to Wooldridge [5], there are four different components in the negotiation process: a negotiation set (a set of possible proposals), a protocol (legal proposals), and a collection of strategies which agents use to make counter-offers.

Negotiation in the SSSHS framework is a one-to-one negotiation, and it proceeds in a series of iterations, with every agent making a proposal at every iteration. Upon reaching the agreement, negotiation terminates with the agreement deal, which is referring to the last counter-offer.

Raiffa's model for bilateral negotiation [228] uses a set of quantitative variables. Let i ($i \in \{a, b\}$) represent agents involved in the negotiation process, j ($j \in \{1, ..., n\}$) represent elements of offer, and $x_j \in [min_j , max_j]$ represent a possible set of values for the element of offer j. Every agent implements a usefulness function $V_j^i : [minj , maxj] \rightarrow [0, 1]$ which assignes value to the individual elements of the offer. The relative weight which the agent i assignes to the offer element j is labeled with w_j^i. The usefulness function of the agent i for an offer $x = (x_1, \ldots, x_n)$ is defined as follows:

$$V^i(x) = \sum_{1 \le j \le n} w_j^i \, V_j^i(x_j) \qquad (2)$$

The SSSHS framework implements two elements of the offer, “quantity” and “worth”. Quantity refers to the amount of resources being negotiated. “Worth” can refer to monetary value, quality of offer, or other factor, decided upon by the modeler. The results of the negotiation process depend on the defined timeout, and the interpretation of the offer. For example, let $a, b \in$ A represent agents in the negotiation process. Agent a has a usefulness function V^a. The interpretation of the offer $x_{b \to a}^t$ of the agent a in time $t’ (t < t’)$ is defined as following:

$$\mathrm{I}^a(t', x_{b \to a}^t) = \begin{cases} refuse \ if \ t' > t_{max}^a \\ accept \ if \ V^a(x_{b \to a}^t) \ge V^a(x_{a \to b}^{t'}) \\ else: x_{a \to b}^{t'} \end{cases} \qquad (3)$$

- $x_{a \to b}^{t'}$ is an offer which the agent a would send to the agent b in time $t’$
- t_{max}^a is the latest time in which the agent a has to end the negotiations

Therefore, the acceptance of the offer is determined by the interpretations of both received and prepared offers, and their mutual comparison.

4.3.2 Upper Threshold Mechanisms

A system may produce **surplus of resources** that its storage unit(s) may not be capable of storing completely at a given time unit, because of its physical limits. In such scenario, a system may experience a loss of produced resources, which could be used (or which could critically lack) in some later simulation time periods. In order to prevent such adverse event, a set of actions is triggered by the system whose common goal is to find a way to advantageously spend the surplus of resources. Such mechanisms are described hereafter.

Upper threshold value is set by the modeler of the system when designing the system, and can be manipulated with subsequently in the efforts to achieve different simulation results.

4.3.2.1 Restoring default consumption modes

This is the first action taken by the framework that is triggered by the upper threshold alert, and it includes calling the consumers that are currently working in the economy mode (previously set by the lower threshold mechanism) and ordering them to restore to their default consumption rates. These default consumption rates are inherently higher than economy mode rates, and as such, they should be consuming more resources, spending the accumulated surplus, and restoring the higher comfort/efficiency values of the consumers. It is possible that there wouldn't be any consumers working in the economy mode at the moment of upper threshold breach, and the restoring mechanism in this case would not proceed.

4.3.2.2 Manipulation of operating times: advancing

If restoring default consumption rates proves to be insufficient, a framework is than able to call currently non-active consumers, and ask them if they could **start their operation now** (now = current timer value), instead of starting in their default operation starting time. Agents could answer with yes or no, depending on their instantiated parameters, which are set by the modeler of the specific system. If the answer is yes, a system is recalculating current demands and production in respect to the current resource level, and bringing a decision if a simulation should proceed to

normal operation, or there might be a need for further interventions in the context of resource overflow.

4.3.2.3 Offering the resources to neighboring sub-systems

If the surplus of resources can't be spent inside the dwelling sub-system, this surplus is offered to the neighboring sub-systems, starting with the one with the lowest resource level. A specific amount to offer is calculated according to the simple formula:

$$Offer = Current\ Resource\ Level - Upper\ Threshold + Minimal\ Unit \quad (4)$$

A surplus of resource is defined as the value of resource which is above the upper threshold. "Minimal Unit" is the smallest amount of resource unit which is computable in the simulation, and it places the new amount of resource, if given away completely, beneath the upper threshold value, preventing the threshold alarm from triggering the upper threshold mechanisms in the same time unit.

The surplus is offered to other dwelling units in the system until it is spent completely. A loss of resources occurs if there is still overwhelming amount of resources left after these actions:

$$Loss\ of\ resources = Current\ Resource\ Level - Maximum\ Storage\ Capacity \quad (5)$$

A global and individual losses of resources are reported back to the modeler at the end of the simulation, pointing to the possibility of insufficient storage capabilities of the modelled system.

A dwelling unit which is asked to accept the offer calculates the amount it can take safely (not to trigger its own upper threshold mechanisms, or to lose the given resources):

$$Take = abs(self.upperThreshold - self.currentResourceLevel - Minimal\ Unit) \quad (6)$$

Decision factors involve upper threshold value and current resource level. “Minimal Unit” ensures that the resource level value stays beneath the upper threshold value, preventing the threshold alarm from triggering the upper threshold mechanisms in the same time unit.

4.4 The SSSHS Environment

The real-world environment in which such agents might exist is directly related to the number and capacities of resource production units system-wide. Spatial/geographical location in which the dwelling network exists can correlate to the availability of fresh water (precipitation, lakes, streams, etc.), sunshine hours and intensity, soil fertility, wind force, potential food locations and varieties, and other resources.

In the proposed model, because of its real-time complexity, the environment will be abstracted into the parameters of the agent classes, and their abilities to produce resources. More specifically, the system is not modeled with sources of various resources in mind (such as the wind, the sun, the rain, etc.) which “exist” in the environment, but with a specific capabilities of the system to catch and store those resources (wind turbines, solar collectors, rainfall collection surfaces, storage tanks, etc.). If there is a lot of sunshine hours in the environment, but the system is not equipped with a proper equipment for utilizing the sunshine (such as solar collectors), then the system has no use of all these sunshine hours, and thus makes this resource irrelevant for the simulation.

Spatial distance between individual dwellings in the proposed dwelling network is relevant to the simulation. It is assumed that resource transfers between individual dwelling units take place between those dwelling units that are closest to one another. Because the proposed model abstracts the possible real world scenarios, it is assumed that in some cases a certain loss of resources happens because of the transfer itself, and by minimizing the transfer distance, the resource loss is minimized as well.

4.5 Framework Goals

Implementing a goal-directed system behaviour is illustratively described in [5, pp. 23]: "*When we write a procedure in Pascal, a function in C, or a method in Java, we describe it in terms of the assumptions on which it relies (formally, its precondition) and the effect it has if the assumptions are valid (its postcondition). The effects of the procedure are its goal: what the author of the software intends the procedure to achieve.*"

Goals are widely discussed in [7], but from the perspective of the user. This work is focused on the goals from the system's point of view, like presented in [8]. Goals help to identify and focus on the critical aspects of the system requirements. As such, the general goal of the system is to fulfill the user's desires, which in the context of this research could be defined as:

> *After persistently trying to make the modeled system self-sustainable, inform the modeler if the modeled system is self-sustainable; if the modeled system is not self-sustainable, refer the modeler to the relevant problems that beset the self-sustainability.*

4.5.1 Basic Framework Requirements

In order to design a model of a realistic dynamic smart self-sustainable human dwelling system, this basic set of **requirements** needs to be met:

- A system needs to be able to produce, consume, and store resources within its boundaries.
- Resource production, consumption, and storage entities are connected in the system via defined interfaces and communication protocols.
- Such entities are connected to their software agents which can directly manipulate their real-world behavior via defined actuators.
- Such entities are autonomous, have their inner goals (maintaining the defined comfort levels), but also share a collective goal (maintaining self-sustainability), and are able to cooperate in order to achieve a defined collective goal.

- Significant simulation parameters, set by the simulation run according to the resource production, consumption and storage dynamics, are required to be reported in real-time for further analysis.
- A modeler of the system should be able to model and define the specifics of the system according to his/hers needs. These specifics refer to defining a basic set of system units (producers, consumers and storages), their connections, and a number of simulation variables and detailed parameterization.
- After modeling a specific system in this framework and given all the needed input, the simulation can be started, system dynamics observed via verbose parameters' output, and the framework eventually stops the simulation and presents the easy readable results.
- Framework should actively try to maintain the self-sustainability of the system with its developed inner behaviours (self-sustainability mechanisms).
- If the framework fails to maintain the self-sustainability of the modeled system, it should inform the user of the main problems that beset the self-sustainability.

4.5.2 Capturing Goals

"*The goals are identified by distilling the essence of the set of requirements*." [97] According to this proposition and from the previously identified requirements, a set of concise goals, deliberately devoid of the requirement details, can be derived as follows:

1. Enable the modeler to model the specific system.
2. Establish system dynamics according to the modeler's input.
3. Establish connections and communications between system entities.
4. Pursue individual agents' objectives.
5. Maintain self-sustainability of the modeled system (a collective goal).
6. Report on the results of the simulation.
7. Report the relevant factors if self-sustainability is not achieved.

4.6 Defining Agent Roles

An agent in the context of this research is a computer software object with most of the widely-agreed upon characteristics of an autonomous agents; it is self-contained, uniquely identifiable, autonomous, self-directed, social, having a state, goal-oriented, etc. To quote Wooldridge, "*An agent is a computer system that is situated in some environment, and is capable of independent (autonomous) action in this environment on behalf of its user or owner, in order to meet its delegated objectives.*" [5], [125]

The third step in the MaSE methodology suggests using the previously defined goals for the purpose of constructing a form more useful to the multi-agent systems, which is designated as **roles**. As defined by [97], "*roles are the building blocks used to define agent's classes and capture system goals during the design phase*."

Similarly, in the "role model" of the Gaia methodology for agent-oriented analysis and design [200], "*a role can be viewed as an abstract description of an entity's expected function.*"

As such, roles are described with two types of attributes:

1) The *responsibilities* of the role. A purpose of the role is for something to be done, a core functionality to be carried out.
2) *Permissions or rights* that are associated with the role. When carrying out the role, an agent has to legitimately exploit some type and amount of resources.

Agent roles in the context of this research are defined hereafter.

Producer role

- *Description*: Producer agent generates resources according to the user input data distribution and its operating times.
- *Responsibilities*: When current global time is within the operating time, adds production values to the overall resource level.

- *Permissions or rights*:
    - Permission to read and change the value of the currentResourceLevel variable of the Storage Agent.
    - Permission to report on its operating time, capacity and its storage resource level.

Consumer role

- *Description*: Consumer agent consumes resources according to the user input data distribution and its operating times.
- *Responsibilities*: When current global time is within the operating time, subtract consumption values from the overall resource levels.
- *Permissions or rights*:
    - Permission to read and change the value of the currentResourceLevel variable of the Storage Agent.
    - Permission to report on its operating time, capacity and its storage resource level

Storage role

- *Description*: Storage agent stores the resources produced by producer agents and allows the consumer agents to consume them. Communicates with consumer and producer agents. Communicates with other storage agents; negotiates for resource transfer between itself and other storage units.
- *Responsibilities*:
    - Wakes producers and consumers.
    - Allows the consumer and producer agents to change its inner state.
    - Tracks changes of its inner state.
    - Triggers self-sustainability mechanisms based on current events.
    - Initialize communication and negotiation with other storage agents.
    - Transfers resources from and to other storage agents.

- *Permissions or rights*:
    - Permission to initiate communication with producers and consumers.
    - Permission to initiate communication with other storages.
    - Permission to give away resources in certain events.

Observer role

- *Description:* Observes and reports on the simulation relevant data and statistics. Observes and reports on the individual storage states.
- *Responsibilities*:
    - Report relevant simulation data and statistics at the end of the simulation.
    - Report every storage inner state at the beginning of each time unit.
- *Permissions or rights*:
    - Permission to access and to report global variables relevant to the simulation run and to the nature of the modeled system.
    - Permission to access and to report individual variables constructing the inner state of the storage units.

In order to deploy an operational simulation system, agent roles are mapped to agent classes, which are described hereafter.

4.7 Defining Agent Classes

The product of the fourth phase of the MaSE methodology (creating agent classes) is an agent class diagram, and the class diagram of the SSSHS development process is shown in the Figure **11**.

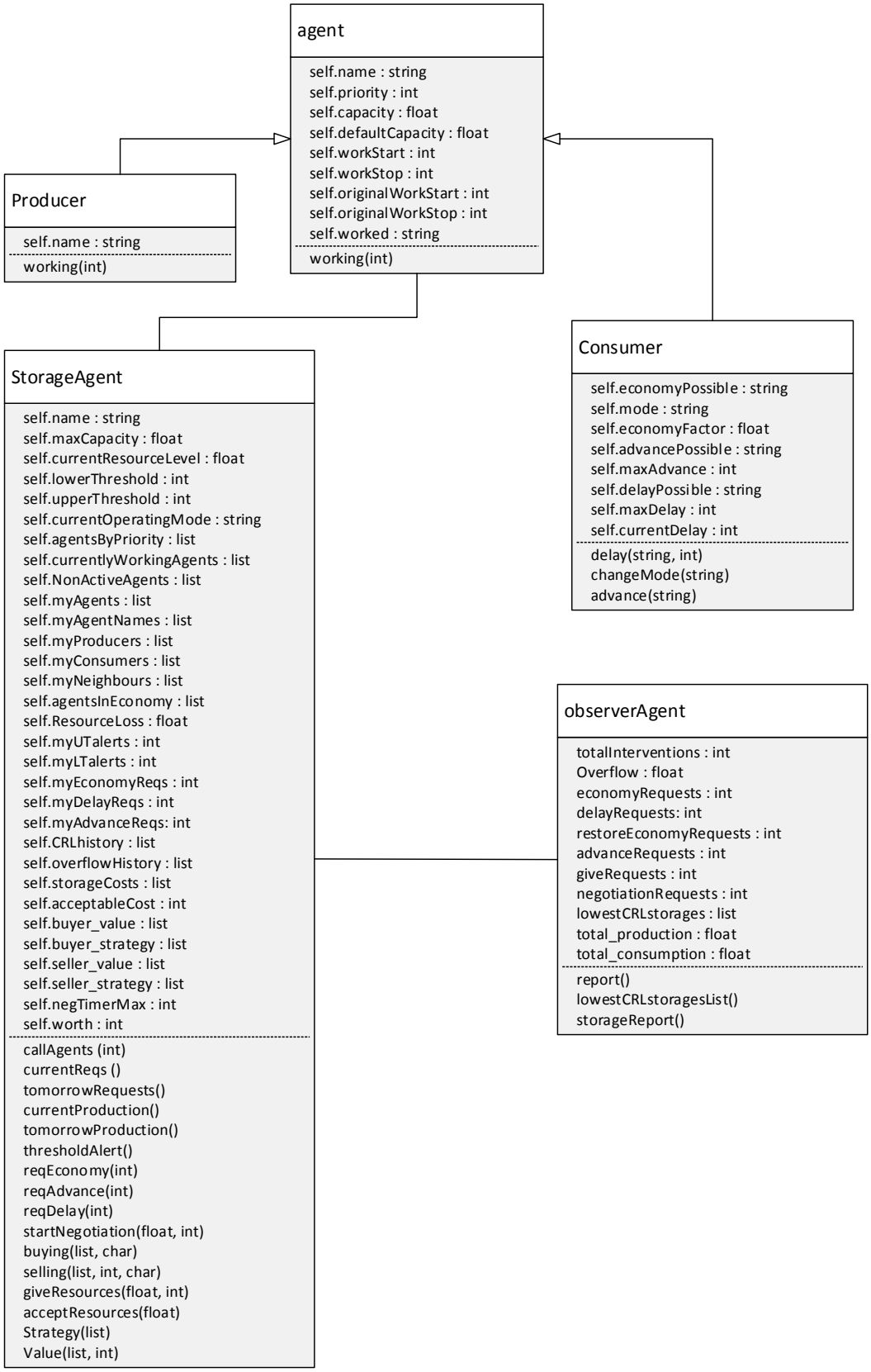

Figure 11. UML class diagram of the SSSHS agent classes

In the SSSHS framework, agents are able to act autonomously, and to collaborate by need. Each agent represents a software agent in a possible physical device in the dwelling subsystem, described through the instantiated parameters of the agent class.

Because of the practical similarity between the Producer and Consumer agent classes, those are implemented as one class, named “agent”, with attribute “capacity” that differentiate them; positive values of the attribute determine a producer, and negative values of the attribute “capacity” determine a consumer. Individual SSSHS agent classes are described hereafter.

4.7.1 Class “agent”

The class “agent” implements both consumers and producer roles. Several variables distinguish them, which is observable at the UML class diagram of the SSSHS agent classes (Figure 11).
Class “agent” directly interacts with the storage unit, either producing, or consuming resources, based on its general type. It can be described by its name, capacity, operating times, self-sustainability mechanisms it supports, etc. In the beginning of each time unit, every active instance of the class “agent” is reporting its internal state and activity.

The class “agent” has a relative priority value, which is used by the SSSHS framework’s load shedding mechanism, where those instances which have lesser priorities are first to be contacted for either a postponed, or a reduced consumption mode.

4.7.2 Class “StorageAgent”

As argued in the reasoning about the feasibility of the agent’s approach in earlier chapters, each unit in a settlement has its own autonomous agent subsystem and knowledge bases, which tries to maintain self-sustainability of the unit.
The Storage Agent class provides the main mechanism for implementing the subsystem’s autonomy (maintaining its own inner state, executing independent decision-making, etc.) and transparentness of its operation (through a number of reporting variables). Actions that are taking place inside each settlement unit are managed by an agent subsystem, more precisely by the methods residing in Storage Agent class.

At certain events, a storage agent initiates a communication with other agents allocated to and managing other units, distributing data to them, and requesting for resource transfer, which would trigger the actuators of the currently hypothetical physical resource network infrastructure, initiating a direct resource transfer.

4.7.3 Class "observerAgent"

The main goal of the Observer Agent is to capture and report relevant statistical data regarding the simulation run. It has several functions:

- Provide both global and individual statistical reports at the end of the simulation. These reports serve as a main reference to the insights regarding quality of the self-sustainability property, to pinpoint possible problems with the self-sustainability (for example, why the self-sustainability can't be achieved), to present data about the loss of resources during simulation run, etc.

- Maintain a list of storage systems sorted by their resource level values

A list of storages refers directly to the settlement sub-units (individual dwellings), and their up-to-date data on the resource levels. The list is being used by the mechanisms supporting self-sustainability for an optimum resource allocation.

- Provide storage data reports at the beginning of every time unit

At the beginning of every time unit, an observer gives insight to the current state of the simulated settlement in the simulation run. The aim is to provide the modeler a relevant insight to the dynamics of the simulation, which could be used to better understand and improve the model of the SSSHS.

4.8 Framework Protocol

Proposed framework model has several key decision points which direct simulation run paths and outcomes. The following diagram (Figure 12) presents an abstract of the proposed system dynamics and illustrates a set of rules which define the nature of communication between agents in the model. Within the SSSHS framework, communication between agents is initiated within specific agent's methods. According to [227], "*the most basic communication method of agents is a procedure call, where the message is encoded within the parameters and the answer is the return value of the procedure.*" [227, pp. 18] Although this communication method is bound to relatively simple communications, it provides adequate grounds for the SSSHS framework's designed functionality. Some of the key message exchanges defined by the protocol are presented hereafter.

- **reqDelay (***lowConsumer***,** *delayTime***).** This message exchange occurs between storage and consumer agents via "reqDelay" method.
  "*lowConsumer*" refers to the specific consumer agent which is targeted for this procedure call based on its relative priority. Example of this parameter is an agent object reference such as "<__main__.agent object at 0x0000000002BAA668>".
  "*delayTime*" is a parameter relating to the exact number of time units which define requested time delay of the targeted consumer agent. This parameter can be defined as integer.

- **reqAdvance (***lowConsumer***,** *advanceTime***).** This message exchange occurs between storage and consumer agents via "reqAdvance" method.
  "*lowConsumer*" refers to the specific consumer agent which is targeted for this procedure call based on its relative priority. Equivalent to the "reqDelay" method, an example of this parameter is also an agent object reference such as "<__main__.agent object at 0x0000000002BAA668>".
  "*advanceTime*" is a parameter relating to the exact number of time units which define requested time advance of the targeted consumer agent, and can be represented as an integer data type.

- **reqEconomy** (*lowConsumer*). This message exchange occurs between storage and consumer agents via “reqEconomy” method.
  “*lowConsumer*” parameter in this case is equivalent to the “lowConsumer” parameter in previous two methods.

- **startNegotiation** (*neededResources, neighbour, transferCosts*). This message exchange occurs between individual storage agents which belong to the same resource network, via “startNegotiation” method.
  “*neededResources*” parameter refers to the calculated amount of resources requested in this method’s communication process (type int).
  “*neighbour*” parameter refers to the storage unit which is being contacted according to defined framework’s protocol rules (for example, according to the physical distance between storage units and/or costs assigned to specific resource tranfers). This parameter is represented as an agent object reference such as “<__main__.agent object at 0x0000000002BAA668>”.
  “*transferCosts*” refers to the assigned costs of resource transfer between individual storage units (type int).

- **giveResources** (*surplus, neighbour*). This message exchange occurs between individual storage agents which belong to the same resource network, via “giveResources” method.
  “*surplus*” parameter refers to the calculated amount of resource surplus in this method’s communication process (type int).
  “*neighbour*” parameter refers to the storage unit which is being contacted according to defined framework’s protocol rules (for example, according to resource levels in individual storage units in the network). This parameter is represented as an agent object reference such as “<__main__.agent object at 0x0000000002BAA668>”.

- **buying** / **selling** (*offer, counter_offer, buyer, seller, timer*). Negotiation process includes two agents; one agent that is asking for services (“buyer”), and other agent that is providing the services (“seller”). The messages that are exchanged in the process are described hereafter.

"*offer*" parameter refers to the offer being sent from the buyer to the seller. It is implemented as a list of two parameters which are being negotiated upon, for example "[50, 120]", where 50 represents a resource quantity, and 120 can for example represent the price for these resources.

"*counter_offer*" parameter is equivalent to the "offer" parameter, sent by the seller following the negotiation protocol.

"*buyer*" and "*seller*" are agent object references, identifying each agent in the negotiation process to the other agent.

"*timer*" parameter quantifies iterations in the negotiation process and synchronizes the agents involved in the process (type int).

A significant element of the SSSHS framework's protocol is depicted in Figure 10. A finite state machine of the negotiating client).

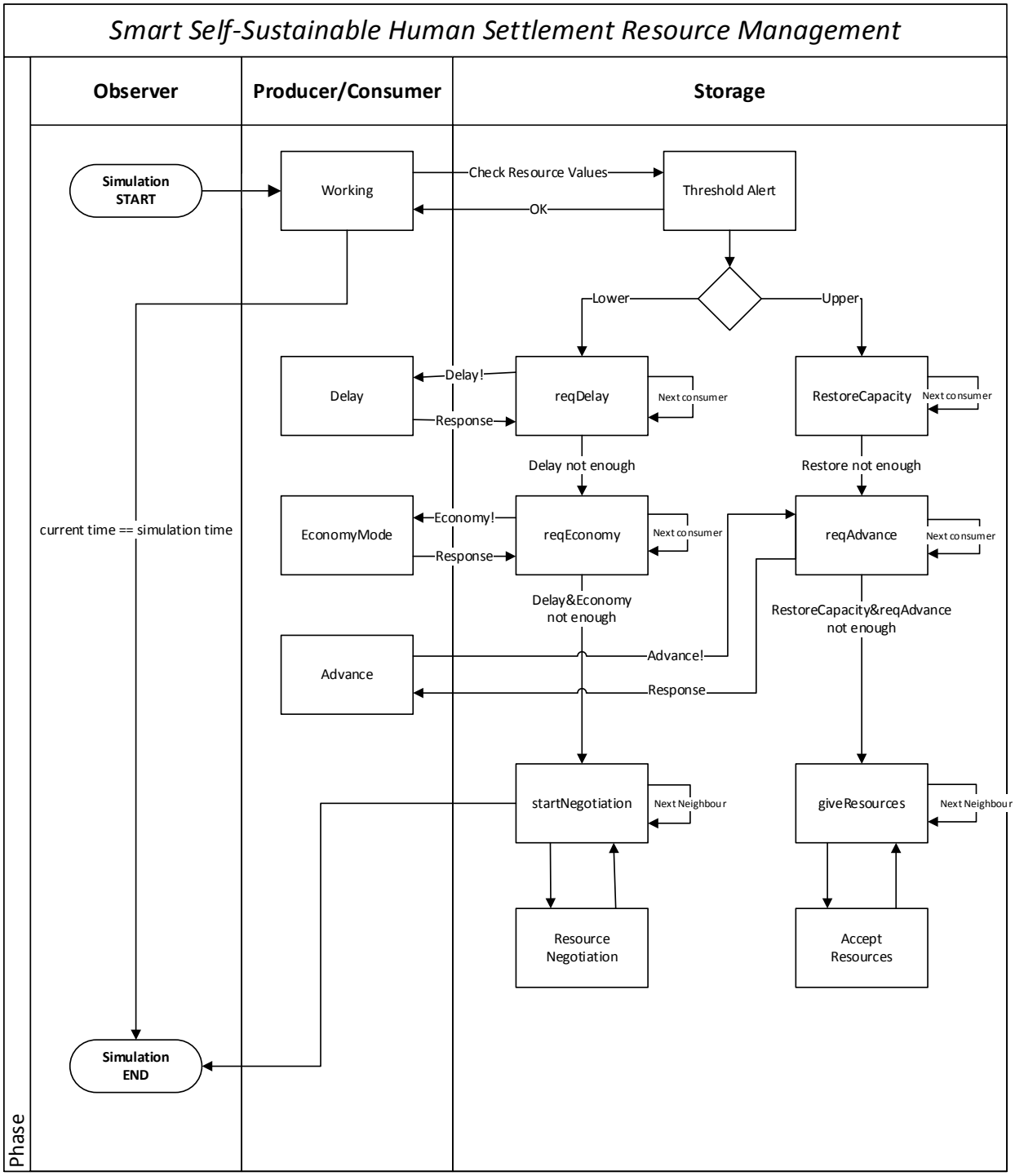

Figure 12. Activity diagram of the SSSHS framework

4.9 The Clock and the Framework's Time

The clock of the framework triggers the set of actions scheduled for a user-defined time periods. Each production/consumption unit has its own internal operating time, and triggers its basic set of action (produce or consume resources) according to the global clock value. If some or more of the self-sustainability mechanisms are triggered in the observed time unit, it is implicitly implied in the framework that they are also resolved in the same time unit, before proceeding to the next time unit. In other words, framework's self-sustainability mechanisms ideally take no time units for their execution.

User/modeler initially sets the simulation runtime before starting the simulation. As time units of the framework are abstracted in order to provide more modelling flexibility, modeler sets this value according to the system's need. For example, if the modeler wants to define a simulation run equal to a 30-day simulation, the variable is set to 30, and other model parameters are set accordingly (input distribution values, operating times, etc.). In this case, 1 time unit in the simulation run refers to 1 day from real world perspective.

4.9.1 Simulation termination

Simulation can be terminated in two ways:

1) Simulation ends as scheduled by the value of the simulation run time, set by the modeler. If this is the case, the simulated system is considered self-sustainable. The framework notifies the modeler of the possible resource losses that took place because of the insufficient storage capacities, and of the number of system interventions that took place in order to avoid the loss of resources.

2) Simulation ends because there was a critical resource deficit, with every self-sustainable mechanism exhausted. The system is proclaimed as being non self-sustainable, and the modeler is notified of the critical limitations of the modelled system which prevent it from being self-sustainable.

If the simulation terminates early in the simulation runtime due to the depletion of resources, an architectural change in the modeled system would be typically recommended. Failures that occur near the end of the simulation runtime could be rectified by model's parameter re-configuration. In any case, a detailed analysis should be performed based on the SSSHS framework's reporting variables.

There are two categories of reporting variables: timer-related and observer-related reporting variables. Timer-related variables report their values at every change of the time unit, where, more specifically, every agent in the simulation run is reporting its inner state at the beginning of each time unit. After the simulation termination executed by either of the two described mechanisms, global statistical variables are presented to the modeler for further consideration and analysis – those variables are labeled as observer-reporting variables.

4.10 Relevant Framework Methods: An Overview

The following methods were identified as relevant and implemented in the SSSHS framework; they were derived from several sources, namely from the reviewed literature on the subjects including intelligent buildings, eco feedback, load management, resource consumption, self-sustainable human settlements; from the informal interviews with three experts on self-sustainability; from the real-world resource distribution mechanisms used by the gas and electric companies; and from the knowledge of the operation of a number of widely used consumption units, personal experiences and analysis.

Method "working"

Producer class is able to "produce" specified quantity of a given resource per time unit. All producer agents are connected to a shared storage agent, and are able to change the inner state of the storage agent by changing the value of a Current Resource Level variable. This variable represents the current amount of available resources. To further clarify this class, it is important to stress that the instantiated producer agent does not represent an independent source of energy, such

as the Sun, but an individual ability to collect this energy, for example a solar collector, and the values it represents are directly available for consumption.
This method also reports its working times, capacity and the changes it has made to its storage unit.

Method “thresholdAlert”

Storage agents are endeavoring to maintain the resource levels between the two threshold limits by the constant oversight of its inner state through this method. If either the pre-determined upper or the lower threshold values are reached in the simulation run, the series of actions are triggered in order to keep the resource levels between those values. Reaching upper threshold could indicate a resource variation trend which could eventually lead to the loss of resources. Reaching lower threshold value could indicate a resource variation trend which could eventually lead to depleting the sub-system of all resources.

Method “reqDelay”

Inside a storage agent, Current Resource Value parameter has upper and lower threshold. If storage resource value equalize with the defined lower threshold, storage agent is sending messages to its consumer agents asking them to postpone their operating times in order to shed consumption load from the system.

Method “reqEconomy”

If the “reqDelay” method does not succeed in the current consumption load shedding, storage agent is sending messages to its consumer agents with instructions to lower their consumption rates by changing their working mode to “economy” (resource savings mode). Consumer with lowest relative priority is contacted first, following by the consumer with next lowest priority, etc., until the resource level in the storage is beyond the lower threshold zone.
Storage agent maintains a list of currently active producers and consumers within its subsystem.

Method “changeMode”

Consumer class is able to decide about whether it will accept the request for change to economy mode of operation, and it notifies the storage unit about its decision. If the decision is positive, i.e. the request is approved, the consumer changes its inner state and sets its current resource

consumption to the value determined by the economy factor set by the modeler of the system. If the request is denied, the consumer notifies the storage about its decision and remains in the current state.

Method “startNegotiation”

If all the available consumers within the subsystem are operating in the “economy” mode and the storage resource level is still equal or less to lower threshold value, the storage agent then **initiates a negotiation for resource transfer** from a neighboring dwelling unit.

Method “Strategy”

Based on the two-parameter input provided by the calling method (*current offer* and *strategy values*), and on the implemented strategy algorithm, this method provides new offer values for the current negotiation iteration.

Method “Value”

Based on the two-parameter input provided by the calling method (*offer values* and *qualitative values*), and on the usefulness function algorithm, it provides the value of usefulness of the input offer.

Methods “buying” and “selling”

A negotiation process is implemented within these methods according to negotiation elements and protocol described in subsection 4.3.1.3.

Method “reqAdvance”

If the Current Resource Level inside the storage agent reaches the upper threshold, the storage agent is requesting from its connected Consumers that they **advance their operating times** in order to expedite the surplus of resources. The assumption is that the storage has its physical boundaries regarding the quantities of resources which it can store.

Method “giveResources”

In the event of overflow of resources, a dwelling unit can contact its neighboring units with the offer to transfer resources to their storages. This mechanism prevents the surplus of resources from being irreversibly lost. In order to optimize this mechanism, the framework is keeping the list of dwelling units sorted by their levels of available resources, with the aim to first contact the unit with least resources available in their storage.

Method “acceptResources”

Associated with the method “giveResources”, this method facilitates the decision-making for the dwelling unit about whether to accept the offered resources from another unit, or to discard the offer. The unit initiates a “diagnostic” if its inner state, calculating whether the offered amount of resources is acceptable. If the offered amount would consequently trigger the upper threshold of the receiving unit, the full amount is declined. Based on the further calculations, it is then possible either to accept a partial amount of offered resources, or to decline the offer completely.

4.11 The Development Environment: Python

Python is a dynamic object-oriented, general purpose and high level programming language that can be used for software development in numerous areas. The language was chosen as an SSSHS framework developing infrastructure because of several reasons:

- it offers strong support for integration with other languages and tools. This might prove useful in using the SSSHS framework with other tools and environments.
- it emphasizes code readability, and can be used to build models with a transparent code. The public nature of this research could use this language feature for interested parties to better review and understand the SSSHS framework.
- it comes with extensive standard libraries, and has a short learning curve. This keeps the focus on the research itself, instead on the instrument.
- it is developed under an OSI-approved open source license, making it freely usable and distributable, as the SSSHS framework would be.
- it runs of numerous architectures: Linux/UNIX, Windows, Mac OS X, Android, etc.
- it is very expressive. That assumes that it can express concepts in fewer lines of code than would be possible in languages such as C++ or Java. [201]
- it enables programming both on a small and large scale, and is designed to be highly extensible. [202]

Amongst numerous practical uses in various fields, Python is also effectively used in the scientific computing, with libraries such as NumPy, SciPy and Matplotlib [203-204], and contains specialized libraries which facilitate domain-specific functionality.

4.11.1 Implementation of the Agent Classes

SSSHS framework's agent classes are implemented in Python programming language, and are presented as such. Variables of the class "agent" are defined in the following listing.

```
class agent:
    def __init__ (self, name, priority, capacity, workStart, workStop,
belongsTo, eFactor):

        agents.append(self)
        agentStorage.update({self:belongsTo})

        self.name = name
        self.priority = priority
        self.capacity = capacity
        self.defaultCapacity = capacity
        self.workStart = workStart
        self.workStop = workStop
        self.originalWorkStart = workStart
        self.originalWorkStop = workStop
        self.mode = "normal"
        self.economyPossible = "YES"
        self.economyFactor = eFactor
        self.advancePossible = "YES"
        self.maxAdvance = 12
        self.delayPossible = "NO"
        self.maxDelay = 6
        self.currentDelay = 0
        self.worked = "NO"
```

Listing 1. Definition of the class "agent" (excerpt)

Attributes of the class “agent” in more detail:

- **name**. A user-friendly identification of the producer/consumer.
- **priority**. Relative priority of the agent compared to other agents. Priority is primarily considered when demanding lower consumption or delayed operation of the connected consumers.
- **capacity**. This is the list of values which are added or subtracted to a current resource levels in the dwelling subsystem, if the current simulation time is within the agent’s operating time. A modeller inputs these values based on the need of the specific model. Capacity values can be decreased in the simulation run if the consumer enters the “economy” mode.
- **defaultCapacity**. The initial, default values of the agent’s consumption value. These values are not prone to change; they are constant throughout the simulation run. The list is accessed when the consumer needs to be returned to the “normal” operating mode.
- **workStart**. Current starting operating time of the agent; can be any integer within the simulation length time interval. Subjected to change during simulation run according to the SSSHS framework rules.
- **workStop**. Current stopping operating time of the agent; can be any integer within the simulation length time interval. Subjected to change during simulation run according to the SSSHS framework rules.
- **originalWorkStart**. The initial, default value of the agent operating starting time; used to restore the current operating time to default value.
- **originalWorkStop**. The initial, default value of the agent operating stopping time; used to restore the current operating time to default value.
- **mode**. If the mode is set to “normal”, the agent is consuming/producing resources according to the default values stored in the “capacity” list. If the mode is changed to “economy” during simulation run, the agent is using capacity values decreased by the economy factor.
- **economyPossible**. If set to “YES”, the agent can (by request) decrease its consumption values by the economy factor determined by the modeler. If set to “NO”, the agent is not capable of operating in a mode with less resource consumption than set by the default.

- **economyFactor**. A number from range 0.1 - 0.9 by with the default capacity values are multiplicated when agent enters its savings mode. This number is defined and entered by the modeler in the system instantialization.
- **advancePossible**. Determines if it possible to advance the operating times of consumers in order to spend the surplus of resource If there is more resources produced than it is possible to store locally in a dwelling subsystem.
- **maxAdvance**. Maximum value of advancing operating time, set by the modeler.
- **delayPossible**. Determines if it is possible to postpone the operating time of a consumer in order to avoid the resource deficiency or peak consumption values.
- **maxDelay**. Maximum time units for which the time delay of a consumer is possible.
- **currentDelay**. If consumer's operating time is postponed by request, this variable keeps the track of the cumulative postpone time.

The variables of the class “StorageAgent” are defined in the following listing.

```
class StorageAgent:
    def __init__ (self, name, crl, maxCapacity, lowerThreshold, upperThreshold):

        storages.append(self)

        self.name = name
        self.maxCapacity = maxCapacity
        self.currentResourceLevel = crl
        self.lowerThreshold = lowerThreshold
        self.upperThreshold = upperThreshold
        self.currentOperatingMode = "Normal" #normal, economy, or delayed
        self.agentsByPriority = []
        self.currentlyWorkingAgents = []
        self.NonActiveAgents = []
        self.myAgents = []
        self.myAgentNames = []
        self.myProducers = []
        self.myConsumers = []
        self.myNeighbours = []
        self.agentsInEconomy = []
        self.ResourceLoss = 0
        self.myUTalerts = 0
        self.myLTalerts = 0
        self.myEconomyReqs = 0
        self.myDelayReqs = 0
         self.myAdvanceReqs = 0
        self.CRLhistory = []
        self.overflowHistory = []
        self.storageCosts = storageCosts
        self.acceptableCost = acceptableCost
       self.buyer_value = buyerValue
       self.buyer_strategy = buyerStrategy
       self.seller_value = sellerValue
       self.seller_strategy = sellerStrategy
       self.negTimerMax = negTimerMax
       self.worth = worth
```

Listing 2. Definition of the class "StorageAgent" (excerpt)

Attributes of the class “StorageAgent” in more detail:

- **maxCapacity**. Maximum storage capacity of the storage.
- **currentResourceLevel**. Current level of resource quantity. This is a shared, highly dynamic variable prone to continuous changes made by the producer and consumer agents within the system.
- **upperThreshold**. A value which triggers alert about possible resource overflow. It also triggers specific actions in order to expedite the surplus of resources. If current level of resources eventually go beyond the maxCapacity value, they are effectively lost.
- **lowerThreshold**. If current resource value equalizes with value stored in lowerThreshold, an alert is triggered about possible resource deficit, and specific set of actions are taken in order to prevent the loss of a resource.
- **agentsByPriority**. A list of agents sorted by their relative priority. It is always used when requesting a delayed or reduced consumption from the consumers.
- **currentlyWorkingAgents**. A list of agents which have their operating time values within current global time set by the timer variable. This list is used for example by the storage agent when requesting a delayed or reduced consumption from the consumers.
- **NonActiveAgents**. A list of agents which have their operating time values greater than current global time set by the timer variable. This list is used by the storage agent when requesting an earlier start from the consumers in order to help expedite the surplus of resources.
- **myAgents**. A list of all agents that belong to the specific storage.
- **myNeighbours**. A list of neighbouring dwelling subsystems and their relative spatial distances. These distances are taken into account when contacting the neighbors for a resource negotiation and transfer.
- **agentsInEconomy**. A list of all agents currently working in an economy mode. This list is accessed by the storage when calling the consumers to restore to their default consumption values stored in the defaultCapacity variable, in order to help expedite the surplus of resources.
- **overflowHistory**. A list of resource overflow loses per time unit.
- **storageCosts**. Resource transfer costs to and from other storage units.

- **acceptableCost**. A cost of resource transfer which is considered as acceptable for the transfer to commence. Should the value of the cost be higher than the one defined in this variable, the process of resource transfer will not commence.
- **resourceID.** Identification of the resource type which the storage is processing. Resource transfer is possible between the storages that have the same resource identification value.
- **buyer_value**. A list of qualitative values required for the usefulness function in the negotiation process from the buyer (client) perspective.
- **seller_value**. A list of qualitative values required for the usefulness function in the negotiation process from the seller (server) perspective.
- **buyer_strategy**. A list of values needed for the strategy method in order to produce a new offer based on the strategy algorithm, from the buyer (client) perspective.
- **seller_strategy.** A list of values needed for the strategy method in order to produce a new offer based on the strategy algorithm, from the seller (server) perspective.
- **negTimerMax**. A value that determines maximum possible number of iterations in the negotiation process.
- **worth**. A second element of the offer in the negotiation process. Can be assigned to monetary value, quality, or other aspects, as defined by the modeler of the system. The variable defines initial level of the chosen offer element.

Class “observerAgent” enables a detailed analysis of the simulation run based on global reporting variables. It is defined in the following listing.

```
class observerAgent:

    def report (self):

        totalInterventions = economyRequests + delayRequests + restoreEconomyRequests +
advanceRequests + giveRequests + negotiationRequests

        global Overflow
        for s in storages:
            Overflow += s.ResourceLoss

        print ("\n\n .... [ END OF SIMULATION ] ....")
        print ("\n\n ******* Number of system interventions: %d" %totalInterventions)
        print (" *********** First intervention happened at time: %d"
%firstIntervention)

        print ("\n\n ******* Number of LT ALERTS: %d" %LTalerts)
```

```
        print ("\n *********** Number of DELAY  requests: %d" %delayRequests)
        print (" *********** Number of ECONOMY requests: %d" %economyRequests)
        print (" *********** Number of NEGOTIATION requests: %d" %negotiationRequests)

        print ("\n\n ******* Number of UT ALERTS: %d" %UTalerts)
        print ("\n *********** Number of RESTORE requests: %d" %restoreEconomyRequests)
        print (" *********** Number of ADVANCE requests: %d" %advanceRequests)
        print (" *********** Number of GIVE requests: %d" %giveRequests)
        print (" *********** Overflow of resources: %f" %Overflow)

        for s in storages:
            print ("\n\nINDIVIDUAL REPORT FOR STORAGE %s" %s.name)
            print (" - CRL: %d" %(s.currentResourceLevel))
            print (" - UT alerts: %d" %(s.myUTalerts))
            print (" - Advance reqs: %d" %(s.myAdvanceReqs))
            print (" - Resources lost: %f" %s.ResourceLoss)
            print (" - LT alerts: %d" %(s.myLTalerts))
            print (" - Economy reqs: %d" %(s.myEconomyReqs))
            print (" - Delay reqs: %d" %(s.myDelayReqs))
            print ("CRL HISTORY: %s" %s.CRLhistory)

        exit()

    def lowestCRLstoragesList (self): # a list of storages sorted by their current
resource values

        lowestCRLstorages = sorted(storages,
key=operator.attrgetter("currentResourceLevel"))
        return lowestCRLstorages


    def storageReport (self):

        for s in storages:

            total_production = s.currentProduction()
            total_consumption = s.currentReqs()

            print("\n Hello, I am STORAGE %s, regularly reporting:" %(s.name))
            print(" ---- My resource level: %f" %s.currentResourceLevel)
            s.CRLhistory.append(s.currentResourceLevel)
            print(" ---- My CRL history: %s" %s.CRLhistory)
            print(" ---- My max capacity: %d" %s.maxCapacity)
            print(" ---- My total production: %f" %total_production)
            print(" ---- My total consumption: %f" %total_consumption)
            #print(" ---- My agent codes: %s" %s.myAgents)
            #print(" ---- My agents: %s" %s.myAgentNames)
            print(" ---- My PRODUCERS: %s" %s.myProducers)
            print(" ---- My CONSUMERS: %s" %s.myConsumers)
            print(" ---- My agents working in economy mode: %s " %s.agentsInEconomy)
```

Listing 3. Definition of the class „observerAgent“

The complete description of the reporting variables presented by the class “observerAgent” is available at section 5.2.

5. Using the SSSHS Framework

SSSHS framework can be used by the modeler to model, simulate, and analyze the self-sustainable system. The self-sustainable system can describe for example human settlements based both on and off-Earth, a fleet of independent space vehicles, underwater cities, etc. The user interaction with the framework is enabled through the following key steps:

- Defining the temporal granularity of the simulation and setting the time duration of the simulation run.
- Instantiating the producer, consumer, and storage agents according to the scenario context.
- Defining the inner states of the agents and tuning their behaviours by setting their relevant parameters according to the problem domain and available data.
- Implementing a multi-agent system comprised of a network of agents by defining their mutual relationships.
- Setting the input from the environment affecting the agents and their behaviours.
- Starting the simulation and analyzing the reports produced by the framework at the end of the simulation.

A modeler can change, eliminate, add or reconfigure the building blocks of the modeled system with minimal effort, which can be useful in optimizing the system according to the modeller's goals. A complete SSSHS framework modeling workflow is presented in Figure 13.

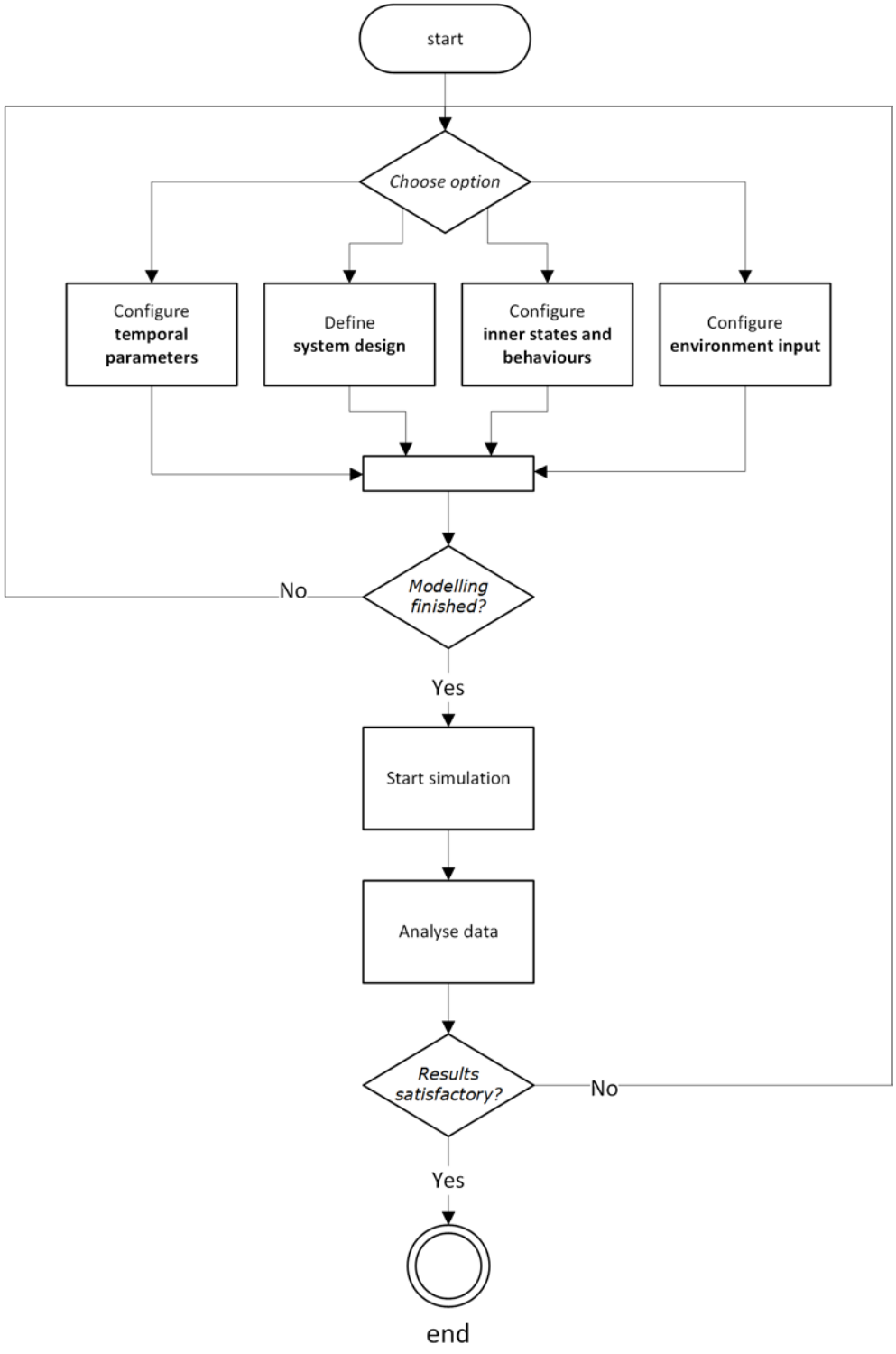

Figure 13. SSSHS framework modeling workflow

5.1 Modelling the System: User Input

Initial use of the framework starts with a model of a specific system that needs to be designed and/or validated against the self-sustainability property. A modeler of the system defines/instantiates specific agent objects from the available classes offered by the framework.

Before modelling, a modeler decides on which resource type a system is going to be tested on. A system is modelled by instantiating agent classes and setting their parameters; connecting producers, storages, and consumers into dwelling subsystems; and connecting these dwelling subsystems into a settlement system which will be simulated for self-sustainability for a given resource type.

Simulation is running for a defined period of time. During the simulation, inner states of the agents are changing, messages are being sent across the system and inside the subsystems, negotiations are taking places, and resource levels are constantly fluctuating inside the subsystems. The goal of the system is to maintain the needed resource levels during the simulation run. If there would be a need for external resources at any given moment during the simulation run, the system would be considered as non-self-sustainable for a measured resource.

The list of more relevant parameters needed to be defined by the modeler for the class “agent” is presented hereafter.

- **name**. A symbolic name of the producer/consumer agent.
- **priority**. Within the specific list of the consumers that shall be within the model, a modeller decides on the relative importance of these consumers against themselves.
- **capacity**. A modeler should have data about the consumer’s/producer’s capability of consuming resources in some time unit.
- **workStart**. A modeler inputs the producer/consumer operation starting time point.
- **workStop**. A modeler inputs the producer/consumer operation ending time point.
- **economyPossible**. Having the real-life scenario in mind, a modeler inputs whether the specific consumer has or has not the ability to operate in a reduced-consumption mode.

- **economyFactor**. A number from range 0.1 - 0.9, representing the factor by which the default capacity values are multiplicated when agent enters its savings mode. This number is defined and entered by the modeler in the system instantialization.
- **advancePossible**. If earlier starting is possible for a specific consumer, a modeler enters “YES”.
- **maxAdvance**. Maximum value of advancing operating time, set by the modeler. A modeler should carefully consider this value, as it affects both the starting and ending operation times of the consumption unit.
- **delayPossible**. Determines if it is possible to postpone the operating time of a consumer in order to avoid the resource deficiency or peak consumption values.
- **maxDelay**. Maximum time units for which the time delay of a consumer is possible. A modeler should carefully consider this value, as it affects both the starting and ending operation times of the consumption unit.

The list of more relevant parameters needed to be defined by the modeler for the class “storageAgent” is presented hereafter.

- **name**. A symbolic name of the storage agent.
- **currentResourceLevel**. A modeler can set the initial level of resources in the specific storage unit. A simulation shall start with this value as a starting point.
- **maxCapacity**. Maximum storage capacity of the storage. A modeler should use the measuring unit consistently throughout the modelling process.
- **upperThreshold**. A modeler sets the “alarm” threshold value of the resource surplus.
- **lowerThreshold**. A modeler sets the “alarm” threshold value of the resource deficit.
- **storageCosts.** Resource transfer costs to and from other storage units. Modeler should define these costs prior to executing the simulation, according to the developed infrastructure network of the modelled settlement.

- **buyer_value, seller_value, buyer_strategy, seller_strategy, negTimerMax, worth**. Variable values are set according to their descriptions presented in subsection 4.11.1.

From the global system perspective, a modeler inputs simulation run time. The time unit is arbitrary; a modeler for example can enter 24, if the target simulation run is one day; or, 30, if the target simulation run is one month; etc.

A modeler also inputs time-distributed values of consumption and production for each unit. Number of distribution values needs to be in correlation with the simulation run time, and presented as list values. For example, the following list represents precipitation data for a 30-day period which is used by the production unit in the simulation run:

```
productionDistribution =
[1.7,19,0,2.3,1.9,0,0.3,0,0,0,0,0,16.7,36.6,6.5,0,0,0,0,29.1,0,0,0,0,2.
5,0,0,12.5,4.3,0]
```

Listing 4. Defining production distribution for a simulation run

5.2 Runtime reporting variables

Runtime reporting provides an informative insight into the dynamics of the simulation at every time unit and major event. Specifically, a framework reports the state of the simulation upon these events:

1) Every change of the timer value (a change of time unit)
2) Resource value reach a predefined upper threshold value
3) Resource value reach a predefined lower threshold value
4) Triggering of the self-sustainability mechanisms
5) Simulation termination

5.2.1 Timer-related reporting variables

These variables report their values at every change of the time unit. More specifically, every agent in the simulation is reporting its inner state at the beginning of each time unit. The Listing 5 presents an example of the storage unit reporting its inner state at the start of the time unit. The report includes storage identification, current resource level, its maximum storage capacity, total production and consumption calculated at the reporting time unit, and information about the producer and consumer agents attached to it.

```
Hello, I am STORAGE UNIT-1, regularly reporting:
 ---- My resource level: 334.500000
 ---- My max capacity: 2000
 ---- My total production: 915.000000
 ---- My total consumption: -300.000000
 ---- My agent codes: [<__main__.agent object at 0x0000000002B22630>,
<__main__.agent object at 0x0000000002A31048>, <__main__.agent object at
0x0000000002B22668>, <__main__.agent object at 0x0000000002A312B0>,
<__main__.agent object at 0x0000000002B224E0>, <__main__.agent object at
0x0000000002B22518>, <__main__.agent object at 0x0000000002B22550>,
<__main__.agent object at 0x0000000002B22588>, <__main__.agent object at
0x0000000002B225C0>, <__main__.agent object at 0x0000000002B225F8>]
 ---- My PRODUCERS: ['RAINFALL-U1', 'HANDWORK-U1']
 ---- My CONSUMERS: ['IRRIGATION-U1', 'VA-U1', 'Person1-U1', 'Person2-U1',
'Person3-U1', 'Person4-U1', 'Person5-U1', 'Person6-U1']
 ---- My agents working in economy mode: []
```

Listing 5. Regular storage unit report

The framework Listing 6 presents the example of regular reports from producers and consumers. It includes their operating times, attached storage (which identifies their dwelling unit), their working capacity and the changes made to the storage resource levels.

```
> > > PRODUCER RAINFALL-U1 REPORTING
------ WORKING FROM 1 AND 30
------ MY STORAGE: UNIT-1
------ MY CAPACITY: 855.000000
------ NEW RESOURCE LEVEL for UNIT-1: 1189.5

> > > CONSUMER Person1-U1 REPORTING
------ WORKING FROM 1 AND 30
------ MY STORAGE: UNIT-1
------ MY CAPACITY: -50.000000
------ NEW RESOURCE LEVEL for UNIT-1: 1199.5
```

Listing 6. Regular producer/consumer unit report

5.2.2 Observer Variables: System Interventions and the quality/scale of the modeled system's self-sustainability

The framework allows the modeler to observe the behaviour of the model. Self-sustainability property can be measured and the quality and the degree of modeled system's self-sustainability can be determined through several observer statistical variables. These variables are reset in every simulation run, and are being calculated throughout the simulation run.

After the simulation has been terminated by either of the two described mechanisms, these statistical variables are presented to the modeler for further consideration and analysis. These variables are elaborated in more detail hereafter.

Overflow. Default value is „NO“. If the variable is set to „YES“ during a simulation time, it points out that there has been a surplus of produced resources (in certain time unit) that could not be handled with current storage configuration, and certain amount of resources was lost; the modeler should note that a possible solution to this problem is that the storage in the system should be enlarged in order to store all the produced resources – and this is considered in the case if other variables show that there was a need for this surplus in another time period of the simulation. Because the self-sustainable mechanisms also include sharing the resources across the settlement network of dwelling units, this variable, when set to “YES”, could also be an indicator of a misconfigured production/consumption architecture - in this case, too many production units. However, if the modeler planned to broaden the consumption rate via additional consumption units in the future system upgrades, conclusions based on this variable value should be postponed until a final architecture is set.

totalInterventions. A total number of all intervention mechanisms in the system which activated in the simulation run in order to maintain the self-sustainability of the model. Logically, the higher is this number at the end of the simulation run, the lower is the quality of the modelled self-sustainable settlement.

firstIntervention. This is a time unit in which the first self-sustainability mechanism activated in the simulation run. In most cases, the lower this time unit, the lower is the quality of the self-sustainability of the modelled system.

interventionTimes. A list of all time units in the simulation run in which some of the self-sustainability mechanisms triggered due to a critical loss, or a surplus, of resources. This list serves both as the metric for a self-sustainability quality, and as the reference for a modeler for easier tracking of the problematic time periods in the simulation run. The larger the list (more problematic time units), the lower the quality of the self-sustainability.

economyRequests. A total number of the requests to the consumers to enter their economy mode of operation, due to the critical loss of resources in the system. The higher the variable value at the simulation end, the lower the quality of the self-sustainability.

delayRequests. A total number of requests to the consumers to delay their operating times if possible, due to the critical loss of resources in the system. The higher the variable value at the simulation end, the lower the quality of the system's self-sustainability.

restoreEconomyRequests. A total number of the requests sent to the consumer units, which imply restoration of their normal capacity modes from the economy modes, due the current overflow of resources which are at risk of permanent loss (insufficient storage capacity). This method is effectively restoring the defined comfort levels. The higher the variable value at the simulation end, the greater is the need for the storage capacity upgrade.

advanceRequests. A total number of the requests sent to the consumer units, which imply earlier operation of the units than initially defined, due the current overflow of resources which are at risk of permanent loss (insufficient storage capacity). The higher the variable value at the simulation end, the greater is the need for the storage capacity upgrade.

giveRequests. A total number of offers sent to other storage units, which imply giving away some amount of resources due the current overflow of resources which are at risk of permanent loss (insufficient storage capacity). The higher the variable value at the simulation end, the greater is the need for the storage capacity upgrade.

negotiationRequests. A total number of requests sent to other settlement sub-systems, informing them of a need for an immediate resource transfer, due to the critical loss of resources in the system. The higher the variable value at the simulation end, the lower the quality of the system's self-sustainability.

UTalerts. A total number of alerts referring to the upper storage threshold breach. The higher the variable value at the end of the simulation, the greater is the need for a storage capacity upgrade.

LTalerts. A total number of alerts referring to the lower storage threshold breach. The higher the variable value at the end of the simulation, the greater is the need for more resource input.

6. SSSHS Simulations and the Developed test-bed Scenarios

Consistent with the research goal (3), the development of nontrivially complex test-bed scenarios for SSSHS resource management is described hereafter. The scenarios include three categories, namely:

1. **Permaculture** (eco village). This category is derived from a real-world case analysis, and includes two scenarios.
2. **Space colony**. The main reference point for developing an off-Earth human space settlement was the Mars One mission.
3. **Space travel**. This scenario considers a fleet consisting of 5 independent, but networked, space vessels, in a prolonged space flight scenario.

All four scenarios were simulated in the SSSHS framework and the results were compared against the research hypothesis H1.

6.1 Scenario 1: Water management in a permaculture-based eco-village

The word "permaculture" was created by Australian ecologist Bill Mollison and his student David Holmgren in the 1970's to describe an "*integrated, evolving system of perennial or self-perpetuating plant and animal species useful to a man. The word is derived from term "permanent agriculture*". [205]

Similar to variations in usage for the term sustainability through time, use of the word permaculture has also varied since 1970's, and Holmgren later expanded the definition to, "*consciously designed landscapes which mimic the patterns and relationships found in nature, while yielding an abundance of food, fibre and energy for provision of local needs.*" [206]

Some other definitions from the permaculture community include:

"*A permaculture system is a system that resembles nature and is based on natural cycles and ecosystems.*" [207]
"*Permaculture is a set of techniques and principles for designing sustainable human settlements…though permaculture practitioners design with plants, animals, buildings and organizations, they focus less on those objects themselves than on the careful design of relationships among them – interconnections – that will create a healthy, sustainable whole.*"
[208]

"*Permaculture is the conscious design and maintenance of agriculturally productive systems which have the diversity, stability, and resilience of natural ecosystems. It is the harmonious integration of the landscape with people providing their food, energy, shelter and other material and non-material needs in a sustainable way.*" [209]

As evident from these definitions, self-sustainability and permaculture are closely related; self-sustainability can be seen as one of the more important goals of permaculture design.

6.1.1 Scenario 1 Detailed

Scenario 1 is based on the real-world case-study of eco-village located in Sisak-Moslavina County, Croatia, Central Europe and Southeastern Europe, with rough coordinates of 45° N, 15° E. The scenario-relevant data was obtained from two sources: (1) on-site, by personally visiting and analyzing the site parameters; (2) by informal interviews of the site residents via email correspondence.

The village has no municipal water supply, and thus completely relies on its own ability to collect and accumulate water for drinking, sanitation, food preparation, bathing, washing and cleaning, small-scale gardening, etc. Average annual precipitation rate in Croatia is 1,162 millimeters [210] making the rainwater collection in most parts of the country feasible.

The village population is fluctuating through the years, with some of the residents permanently inhabiting, and some visiting for a few weeks at the time.

Precipitation data on a daily basis for the selected area was acquired from the Croatian Meteorological and Hydrological institute, Zagreb, Croatia.

The village is consisted of a 3 infrastructurally independent dwelling units (from now on referred to as “Units”), with the following residential setup:

- Unit 1: 5 fluctuating residents, 1 permanent resident; occasional guest residents
- Unit 2: 2 permanent residents
- Unit 3: 2 permanent residents; occasional guest residents

The selected time frame for the scenario simulation is the month September in the year 2012.
The choice for this exact time period is argued as follows:

- The month September 2012 had a precipitation of 133.4mm, which was above average monthly precipitation rate for this area, averaged from 1949-2013 at 86.4mm [211].

- The settlement ran out of water in the first half of the September. This event is considered scientifically curious for the SSSHS, because the main goal of the framework is to prolong the self-sustainability of the settlement.
- August 2012 had a precipitation of 12.4mm, which was assumed to be the main reason for the resource deficit in September. Average precipitation for August is 81.6mm [211].

In the selected scenario time frame, residential setup was as follows:

- Unit 1: 6 residents
- Unit 2: 2 residents
- Unit 3: 2 residents

Residents are obtaining water from a few sources:

- Rainfall collection via 1000 l tanks, used for various purposes
- Local lake (dries occasionally), hand-picked, used mainly for utilities
- Local water spring, hand-picked, used mainly for drinking and food preparation

6.1.2 Rainfall Collection

The total amount of rainwater that can be harvested via dwelling roofs depends on the dwelling roof area, rainfall depth and runoff coefficient. The latter depends on the roof material and design. [212]

Volume of rainwater harvested at a human settlement in one month period can be determined by using monthly rainfall data, the total roof area, and a runoff coefficient in the following formula [213]:

$$VR = \frac{R\ x\ TRA\ x\ Rc}{1000} \quad (7)$$

Where:

- **VR** = monthly volume of rainwater that could be harvested in each settlement (m^3)
- **R** = monthly rainfall in a settlement (mm/m^2/month)
- **TRA** = total roof area in a settlement (m^2)
- **Rc** = runoff coefficient (nondimensional)
- **1000** = conversion factor from liters to m^3

Slightly adapted formula which calculates the volume of rainwater harvested at each dwelling unit is used as follows:

$$VR = R\ x\ TRA\ x\ Rc \quad (8)$$

Where:

- **VR** = total amount of harvested water (liters per time per dwelling unit)
- **R** = rainfall in mm/time, (1 mm = 1 l/m^2)
- **TRA** = total harvesting roof area (m^2)
- **Rc** = efficiency factor/ runoff coefficient of collection surface

The volume of collected water that is available in the water storage units at the measured time is calculated by subtracting the sum of all needs for water in time t and adding the sum of all water collection systems from the current water volume in the storage units, as presented in the formula [214]:

$$CRL(t) = CRL\ (t\text{-}1) + TP(t) - TC(t) \quad (9)$$

CRL(t) = available resource volume (in this case, water) level at time t

CRL(t-1) = initial resource volume (in this case, water) level at time t-1

TP = total production (the sum of volumes of all water harvesting processes at time t)

TC = total consumption (the sum of all water requirements at time t)

Table 1. Characteristics of roof types [212]

Type	Run-off coefficient	Notes
Galvanized Iron Sheets	>0.9	· Excellent quality water. Surface is smooth and high temperatures help to sterilize bacteria
Tile (glazed)	0.6 – 0.9	· Good quality water from glazed tiles. · Unglazed tile can harbor mould · Contamination can exist in tile joints
Asbestos Sheets	0.8 – 0.9	· New sheets give good quality water · No evidence of carcinogenic effects by ingestion · Slightly porous so reduced run-off coefficient and older roofs harbor moulds and even moss
Organic (Thatch, Palm)	0.2	· Poor quality water (>200 FC/100 ml) · Little first-flush effect · High turbidity due to dissolved organic material which cannot easily be filtered or settled out

According to [212], run-off coefficient for a tile roof is between the values 0,6 and 0,9. Because this value could not be precisely determined without a detailed material analysis and calculations, middle value of 0,75 will be considered as a sufficiently precise value for the scenario purposes.

Table 2 presents daily precipitation data, collected by the weather station Topusko, Croatia, in the year 2012. The month of interest is September, and is slightly highlighted.

Table 2. Daily precipitation data, collected by weather station Topusko in the year 2012

	Jan	Feb	Mar	Apr	May	June	July	Aug	Sept	Oct	Nov	Dec
1	0	0	0	1,3	0	0	0	0	1,7	0,3	2,3	10
2	0	0	0	0	0,3	13,1	0	0	19	10	5	5,6
3	0	1	0	0	0,9	1	0	0	0	8,8	2,8	8,4
4	2,5	5,1	0	0	0	0	0	0	2,3	0	0	0
5	1,7	6,1	0	0,7	0	10,1	0	0	1,9	0	0	16,8
6	3,7	0	0	4,5	0	1,2	0	0	0	0	25,8	0
7	0	7,1	0	8,2	18,3	0	15,2	0	0,3	0	3,6	0,6
8	0	1,8	0	8,5	12,1	0	0	0	0	2,4	0	11
9	0	0	0	1	0	0	0	0	0	0	0	8,6
10	0	0,8	0	0	0	13,5	0	0,8	0	0	0	0
11	1,8	5	0	0	0	2,2	0	0	0	7	0	0,7
12	0	9	0,2	6	0	13,7	0	0	0	0,3	0,6	0
13	0	5,6	5,3	0	12,2	19	0	0	16,7	4,4	17,9	0
14	0,3	0	0	0,2	10,6	13	0	0	36,6	6,2	1,5	0
15	0	0	0	0,5	1,2	0	0	0	6,5	0	0	0
16	0	0	0	1,6	0	0	0	0	0	9,7	0	7,2
17	0	0	0	2,3	30	0	0	0	0	8,2	0	0
18	0	0	0	0	0	0	0	0	0	0	1,4	13
19	0	0	0	0	0	0	0	0	0	0	2,4	12,5
20	0	9,5	1,4	0	0	0	0	0	29,1	0	0,2	2,1
21	9,5	2,4	0	2	1	0	0	0	0	0	0,4	0
22	0	0	0	1,1	6,2	0	3,5	0	0	0	0,2	0,8
23	0	0	0	5,8	16,1	0	13	0	0	0	0,7	0
24	8,4	0,1	0	0,3	11,6	0	0	0	0	0	0	0
25	5,7	0	0	8,4	9,8	0	21,4	0	2,5	0	0	0
26	0	6	0	0	0	2,3	0	0	0	0	0	0
27	0	4	0	0	3	0	4	11,6	0	4,7	0	25
28	0	0	0	0	0	0	0	0	12,5	16,7	1,8	11,4
29	0	0,2	0	0	0	0	0	0	4,3	7,1	14,8	0
30	0,1	0	0	0	0	0	0	0	0	6,7	59,4	0
31	0	0	0	0	3,3	0	0	0	0	0	0	0
TOT.	**33,7**	**63,7**	**6,9**	**52,4**	**136,6**	**89,1**	**57,1**	**12,4**	**133,4**	**92,5**	**140,8**	**133,7**

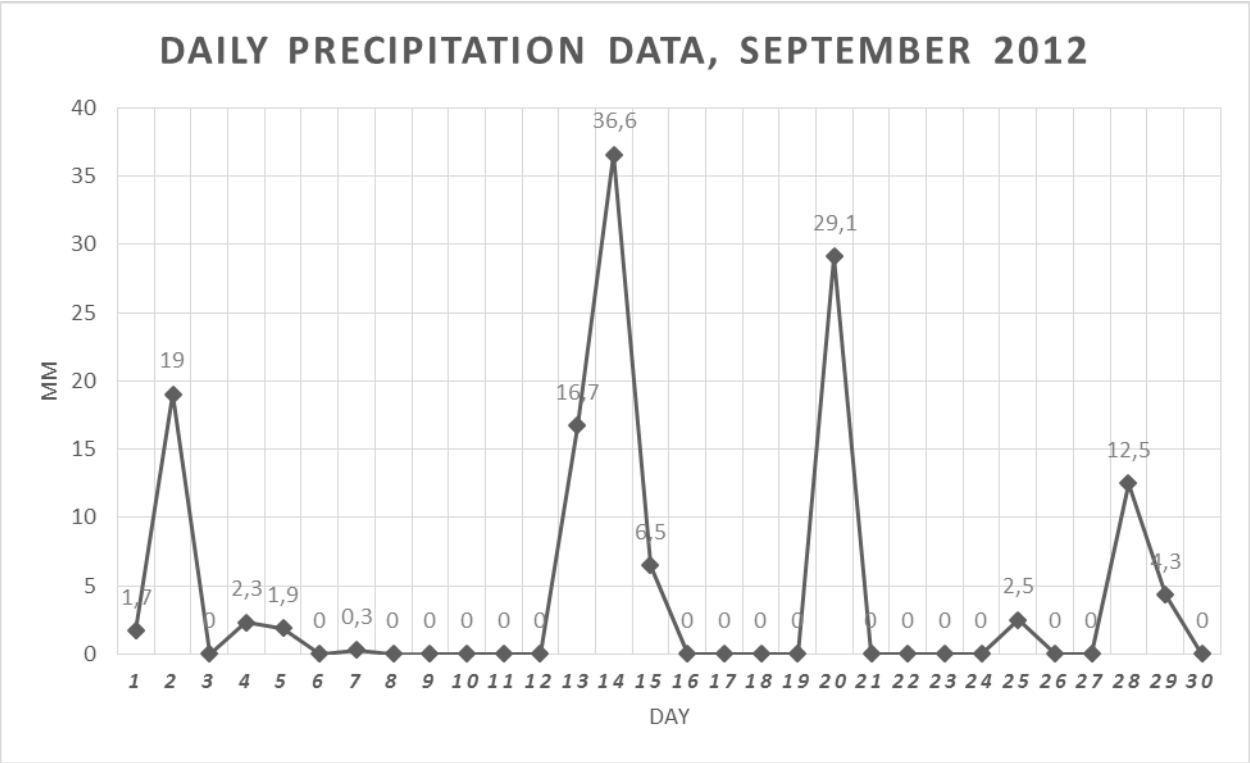

Figure 14. Daily precipitation data, collected by the weather station Topusko in the year 2012

6.1.3 Water harvesting and storing capabilities of the settlement

Water harvesting capabilities of the single dwelling unit depend on several factors such as dwelling's total roof area, roof characteristics, "manual" or automatic water harvesting from wells, lakes, springs, rivers, etc. Storing capabilities of the dwelling unit depend on the number, capacities and material characteristics of the storage units (canisters, cisterns, etc.).

Unit 1

Unit 1 uses its roof surface (60m^2) and two 1000 l cisterns to collect its main amount of used water. Additionally, most days some of the residents are manually collecting water from a local lake and a spring. Initial water level from a previous month is 700 l, and is used as a starting value in the simulation run.

Manually collected water from a lake and a spring with 10 l and 20 l canisters is summarized into one distribution, used as follows:

[60,0,40,0,0,40,60,60,60,60,60,60,0,0,0,60,60,60,60,0,60,40,20,0,60,60,0,60,60,60]

Values are rounded, presuming an ideal water collection precised to the canister capacities. The real values of manually collected water are gravitating around those, and the potentially resulting differences would not present a significant deviations from the simulation results.

Unit 1 uses two main storage units; both of them are cisterns with a capacity of 1000l:

- Cistern 1, storage capacity: 1000 l
- Cistern 2, storage capacity: 1000 l

Total storage capacity for Unit 2 is therefore 2000 l.

Table 3 presents data on daily collected rainwater by Unit 1 in the year 2012, calculated by the formula *(8)*.

Table 3. Daily collected rainwater by Unit 1 in the year 2012

Collected Rainwater: CALCULATED DATA												
	Jan	**Feb**	**Mar**	**April**	**May**	**June**	**July**	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**
1	0	0	0	58,5	0	0	0	0	76,5	13,5	103,5	450
2	0	0	0	0	13,5	589,5	0	0	855	450	225	252
3	0	45	0	0	40,5	45	0	0	0	396	126	378
4	112,5	229,5	0	0	0	0	0	0	103,5	0	0	0
5	76,5	274,5	0	31,5	0	454,5	0	0	85,5	0	0	756
6	166,5	0	0	202,5	0	54	0	0	0	0	1161	0
7	0	319,5	0	369	823,5	0	684	0	13,5	0	162	27
8	0	81	0	382,5	544,5	0	0	0	0	108	0	495
9	0	0	0	45	0	0	0	0	0	0	0	387
10	0	36	0	0	0	607,5	0	36	0	0	0	0
11	81	225	0	0	0	99	0	0	0	315	0	31,5
12	0	405	9	270	0	616,5	0	0	0	13,5	27	0
13	0	252	238,5	0	549	855	0	0	751,5	198	805,5	0
14	13,5	0	0	9	477	585	0	0	1647	279	67,5	0
15	0	0	0	22,5	54	0	0	0	292,5	0	0	0
16	0	0	0	72	0	0	0	0	0	436,5	0	324
17	0	0	0	103,5	1350	0	0	0	0	369	0	0
18	0	0	0	0	0	0	0	0	0	0	63	585
19	0	0	0	0	0	0	0	0	0	0	108	562,5
20	0	427,5	63	0	0	0	0	0	1309,5	0	9	94,5
21	427,5	108	0	90	45	0	0	0	0	0	18	0
22	0	0	0	49,5	279	0	157,5	0	0	0	9	36
23	0	0	0	261	724,5	0	585	0	0	0	31,5	0
24	378	4,5	0	13,5	522	0	0	0	0	0	0	0
25	256,5	0	0	378	441	0	963	0	112,5	0	0	0
26	0	270	0	0	0	103,5	0	0	0	0	0	0
27	0	180	0	0	135	0	180	522	0	211,5	0	1125
28	0	0	0	0	0	0	0	0	562,5	751,5	81	513
29	0	9	0	0	0	0	0	0	193,5	319,5	666	0
30	4,5	0	0	0	0	0	0	0	0	301,5	2673	0
31	0	0	0	0	148,5	0	0	0	0	0	0	0
TOT	**1516,5**	**2866,5**	**310,5**	**2358**	**6147**	**4009,5**	**2569,5**	**558**	**6003**	**4162,5**	**6336**	**6016,5**

Water consumption outline for Unit 1:

1) **Water consumption by residents**. According to Gleick [10], water use for residents is averaged to 50 l/day/person. This is an even distribution, and is used in the simulation as follows:

[-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50]

Table 4. Estimated basic water requirements: NASA and Gleick [128]

Activity	NASA		GLEICK (1996)	
	estimated minimum req't. (l/c/d)	% of sample within minimum	proposed minimum req't. (l/c/d)	% of sample within minimum
Drinking	1.6	97.9	5.0	100.0
Personal Hygiene				
Shower/bathing	2.7	0.0	15.0	34.3
Hand/face washing	4.1	86.4	-	-
Sanitation				
Urinal/toilet flushing	0.5	0.0	20.0	94.3
Cooking & Kitchen				
Food Preparation	0.8	10.0	10.0	96.4
Dish washing	5.5	82.1	-	-
Laundry	12.5	71.9	-	-
Total	**27.7**		**50.0**	

Table 5. Estimated basic water requirements: Mean, Minimum, Maximum [128]

Activity	Mean	Minimum	Maximum
Drinking	0.58	0.30	1.60
Personal Hygiene			
Showering/bathing	19.00	5.70	105.85
Hand/face washing	2.97	0.93	23.10
Brushing of Teeth	1.07	0.40	4.79
Sanitation Services			
Urinal/toilet flushing	7.63	2.38	62.04
Toilet cleaning	1.71	0.50	25.20
House cleaning	1.35	0.27	12.36
Cooking and Kitchen			
Food Preparation	1.87	1.43	2.01
Dish washing	1.96	1.73	2.39
Laundry	4.72	1.90	7.44
Total	**42.86**	**15.54**	**246.78**

2) **Water consumption for irrigation**. Irrigation was performed during dry days' periods (Figure 15). Average water consumption for irrigation was 10 l/m^2. Garden area for Unit 1 is 50m^2 (10m x 5m). Total water requirements per irrigation event is thus 50 x 10 l = 500 l. A distribution for the irrigation requirements is used in the simulation as follows:

[0,0,0,0,0,0,0,0,0,-500,0,-500,0,0,0,0,0,-500,0,0,0,0,-500,0,0,0,0,0,0,0]

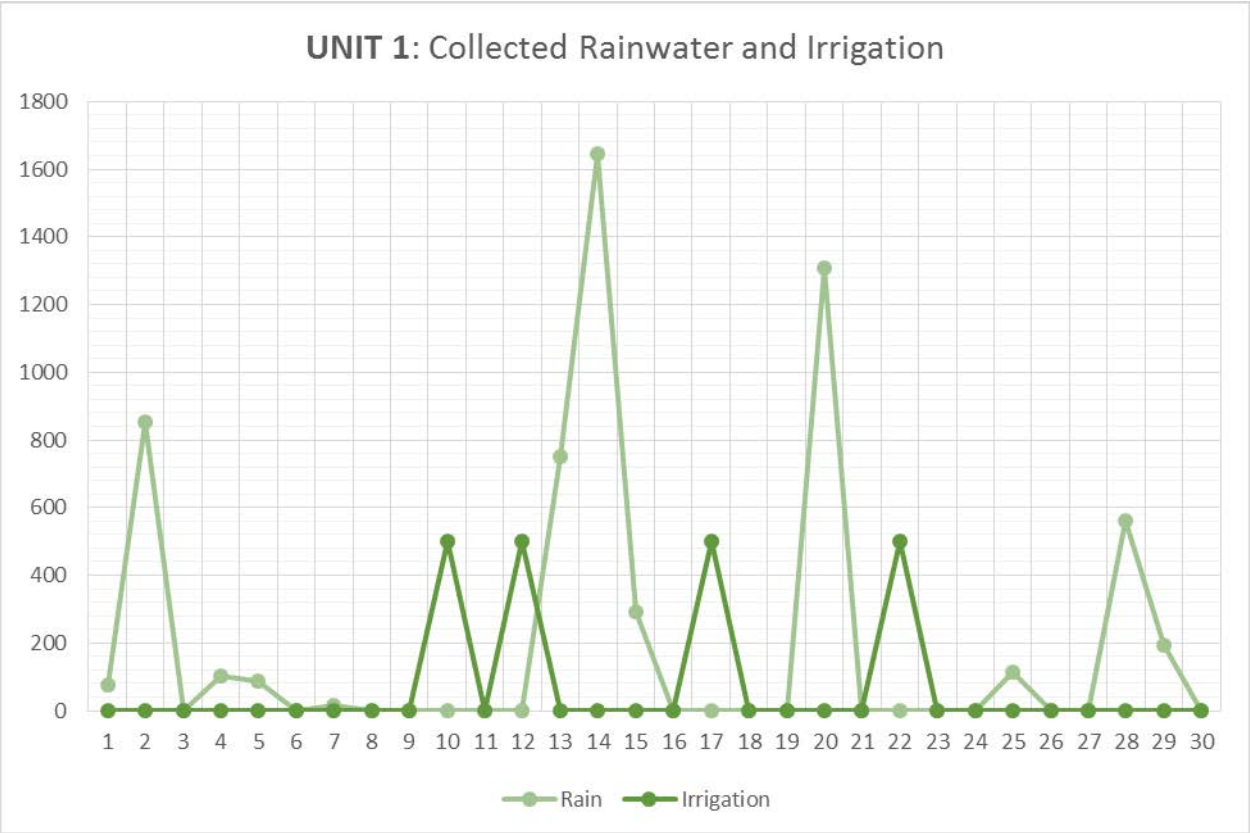

Figure 15. Irrigating the garden at UNIT 1 in zero-precipitation periods

3) **Other, pseudo-random distribution based on occasional and unplanned water needs**. The values were derived from the case-study observations and interviews with the consultants (e.g. washing the equipment, guest accommodation, etc.), and are approximated as follows:

[-2,0,0,-10,0,0,-4,-1,0,-12,0,-1,-1,0,-3,0,0,-21,0,-1,0,-5,-2,-3,0,-7,0,-2,0,-1]

Unit 2

Unit 2 has a roof area of 50m^2. Accordingly, its rainfall collection capacity is lower than it is for Unit 1, as evident from Table 6, which presents data about collected rainwater at Unit 2.

Table 6. Daily collected rainwater by Unit 2 in the year 2012

Collected Rainwater at Unit 2: CALCULATED DATA												
	Jan	**Feb**	**Mar**	**Apr**	**May**	**June**	**July**	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**
1	0	0	0	48,75	0	0	0	0	63,75	11,25	86,25	375
2	0	0	0	0	11,25	491,25	0	0	712,5	375	187,5	210
3	0	37,5	0	0	33,75	37,5	0	0	0	330	105	315
4	93,75	191,25	0	0	0	0	0	0	86,25	0	0	0
5	63,75	228,75	0	26,25	0	378,75	0	0	71,25	0	0	630
6	138,75	0	0	168,75	0	45	0	0	0	0	967,5	0
7	0	266,25	0	307,5	686,25	0	570	0	11,25	0	135	22,5
8	0	67,5	0	318,75	453,75	0	0	0	0	90	0	412,5
9	0	0	0	37,5	0	0	0	0	0	0	0	322,5
10	0	30	0	0	0	506,25	0	30	0	0	0	0
11	67,5	187,5	0	0	0	82,5	0	0	0	262,5	0	26,25
12	0	337,5	7,5	225	0	513,75	0	0	0	11,25	22,5	0
13	0	210	198,75	0	457,5	712,5	0	0	626,25	165	671,25	0
14	11,25	0	0	7,5	397,5	487,5	0	0	1372,5	232,5	56,25	0
15	0	0	0	18,75	45	0	0	0	243,75	0	0	0
16	0	0	0	60	0	0	0	0	0	363,75	0	270
17	0	0	0	86,25	1125	0	0	0	0	307,5	0	0
18	0	0	0	0	0	0	0	0	0	0	52,5	487,5
19	0	0	0	0	0	0	0	0	0	0	90	468,75
20	0	356,25	52,5	0	0	0	0	0	1091,25	0	7,5	78,75
21	356,25	90	0	75	37,5	0	0	0	0	0	15	0
22	0	0	0	41,25	232,5	0	131,25	0	0	0	7,5	30
23	0	0	0	217,5	603,75	0	487,5	0	0	0	26,25	0
24	315	3,75	0	11,25	435	0	0	0	0	0	0	0
25	213,75	0	0	315	367,5	0	802,5	0	93,75	0	0	0
26	0	225	0	0	0	86,25	0	0	0	0	0	0
27	0	150	0	0	112,5	0	150	435	0	176,25	0	937,5
28	0	0	0	0	0	0	0	0	468,75	626,25	67,5	427,5
29	0	7,5	0	0	0	0	0	0	161,25	266,25	555	0
30	3,75	0	0	0	0	0	0	0	0	251,25	2227,5	0
31	0	0	0	0	123,75	0	0	0	0	0	0	0
TOTAL	1263,75	2388,75	258,75	1965	5122,5	3341,25	2141,25	465	5002,5	3468,75	5280	5013,75

Residents are manually collecting water from a local lake and a spring with 10 l and 20 l canisters. These values are varying, and collected water is summarized into one distribution, used as follows:

[0,0,40,0,0,40,20,0,0,20,40,0,0,20,40,40,40,0,0,0,20,0,40,0,40,20,0,20,20,0]

Unit 2 uses one main storage unit - a cistern with a capacity of 1000 l - and a few other smaller storage units:

- Container 1, storage capacity: 200 l
- Container 1, storage capacity: 120 l
- Container 1, storage capacity: 120 l

Total storage capacity for Unit 2 is therefore 1440 l.

Water consumption outline for Unit 2:

1) Water use for residents averaged to 50 l/day/person, used in the simulation as follows:

[-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50]

4) Irrigation was performed during dry days' periods (Figure 16). Average water consumption for irrigation was 10 l/m^2. Garden area for Unit 2 is 30m^2 (6m x 5m). Total water requirements per irrigation process is thus 30 x 10l = 300 l. Final distribution for the irrigation requirements is used in the simulation as follows:

[0,0,0,0,0,0,0,0,0,0,-300,0,0,0,0,0,0,-300,0,0,0,0,-300,0,0,0,0,0,0,0]

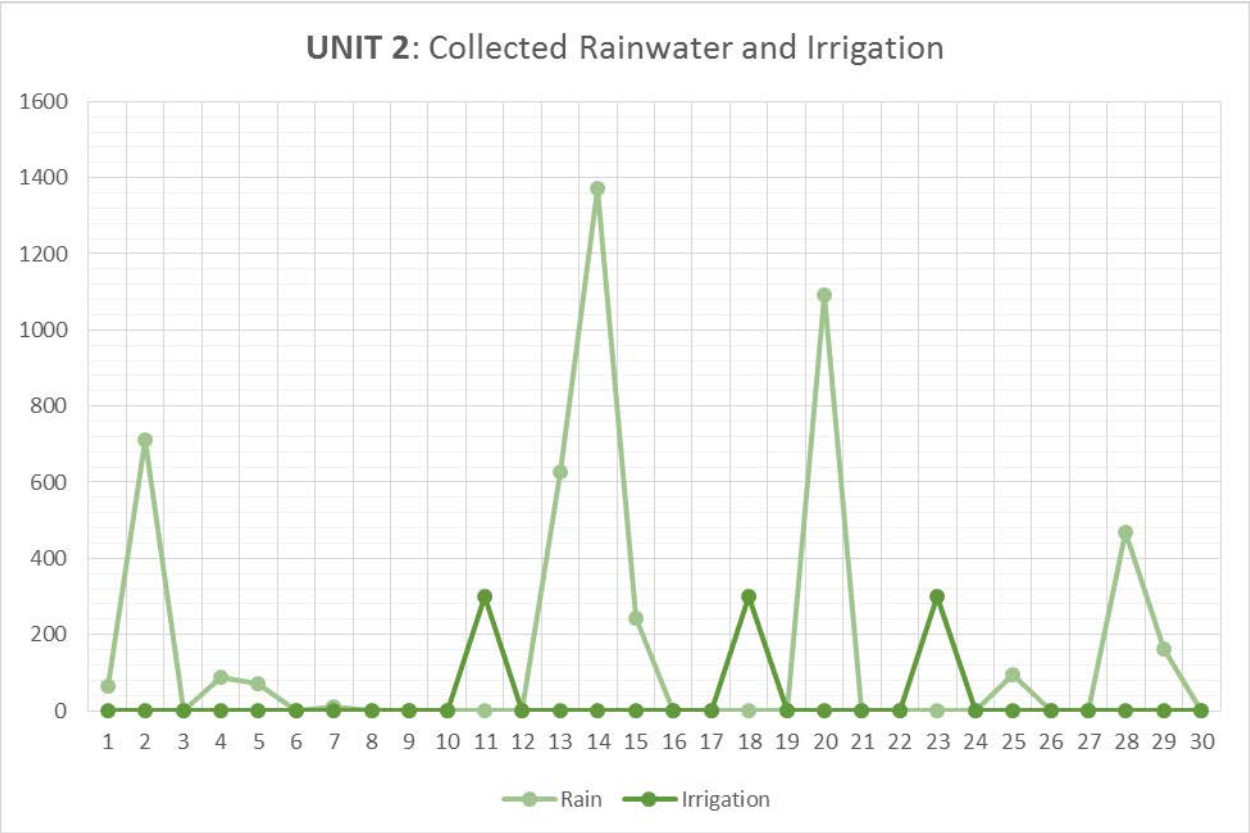

Figure 16. Irrigating the garden at UNIT 2 in zero-precipitation periods

5) Other, pseudo-random distribution based on occasional and unplanned water needs. The values were derived from the case-study observations and informal interviews with the consultants, and are approximated as follows:

[0,0,-25,-70,0,0,0,-8,0,-32,0,0,-14,0,-17,0,0,-0,0,0,-2,0,-28,0,0,0,0,-5,0,0]

Unit 3

Unit 3 uses its roof surface of 72m^2, one cistern of capacity 1000 l and a few smaller storages to collect its main amount of used water. Residents are manually collecting water from a local lake and a spring in only rare occasions, which is used as the following distribution:

[0,0,20,0,0,0,20,0,0,0,20,0,0,0,0,0,0,0,40,0,0,40,0,0,0,0,0,0,20,20,0]

Unit 3 uses one main storage 1000 l unit and a few smaller storages, totaling at 1500 l of storage capacity:

- Cistern 1, storage capacity: 1000 l
- Smaller storages: 500 l

Roof surface of Unit 3 is 72 m^2, making it the largest surface area with highest capacity for rainfall collection. Accordingly, its rainfall collection capacity is higher than it is for both Unit 1 and Unit 2, as evident from Table 7, which presents data about collected rainwater at Unit 3.

Table 7. Daily collected rainwater by Unit 3 in the year 2012

CALCULATED DATA												
	Jan	**Feb**	**Mar**	**Apr**	**May**	**June**	**July**	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**
1	0	0	0	70,2	0	0	0	0	91,8	16,2	124,2	540
2	0	0	0	0	16,2	707,4	0	0	1026	540	270	302,4
3	0	54	0	0	48,6	54	0	0	0	475,2	151,2	453,6
4	135	275,4	0	0	0	0	0	0	124,2	0	0	0
5	91,8	329,4	0	37,8	0	545,4	0	0	102,6	0	0	907,2
6	199,8	0	0	243	0	64,8	0	0	0	0	1393,2	0
7	0	383,4	0	442,8	988,2	0	820,8	0	16,2	0	194,4	32,4
8	0	97,2	0	459	653,4	0	0	0	0	129,6	0	594
9	0	0	0	54	0	0	0	0	0	0	0	464,4
10	0	43,2	0	0	0	729	0	43,2	0	0	0	0
11	97,2	270	0	0	0	118,8	0	0	0	378	0	37,8
12	0	486	10,8	324	0	739,8	0	0	0	16,2	32,4	0
13	0	302,4	286,2	0	658,8	1026	0	0	901,8	237,6	966,6	0
14	16,2	0	0	10,8	572,4	702	0	0	1976,4	334,8	81	0
15	0	0	0	27	64,8	0	0	0	351	0	0	0
16	0	0	0	86,4	0	0	0	0	0	523,8	0	388,8
17	0	0	0	124,2	1620	0	0	0	0	442,8	0	0
18	0	0	0	0	0	0	0	0	0	0	75,6	702
19	0	0	0	0	0	0	0	0	0	0	129,6	675
20	0	513	75,6	0	0	0	0	0	1571,4	0	10,8	113,4
21	513	129,6	0	108	54	0	0	0	0	0	21,6	0
22	0	0	0	59,4	334,8	0	189	0	0	0	10,8	43,2
23	0	0	0	313,2	869,4	0	702	0	0	0	37,8	0
24	453,6	5,4	0	16,2	626,4	0	0	0	0	0	0	0
25	307,8	0	0	453,6	529,2	0	1155,6	0	135	0	0	0
26	0	324	0	0	0	124,2	0	0	0	0	0	0
27	0	216	0	0	162	0	216	626,4	0	253,8	0	1350
28	0	0	0	0	0	0	0	0	675	901,8	97,2	615,6
29	0	10,8	0	0	0	0	0	0	232,2	383,4	799,2	0
30	5,4	0	0	0	0	0	0	0	0	361,8	3207,6	0
31	0	0	0	0	178,2	0	0	0	0	0	0	0
TOTAL	**1819,8**	**3439,8**	**372,6**	**2829,6**	**7376,4**	**4811,4**	**3083,4**	**669,6**	**7203,6**	**4995**	**7603,2**	**7219,8**

Water consumption outline for Unit 3:

1. Water use for residents is averaged to 50 l/day/person, and is used in the simulation as follows:

[-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50]

2. Irrigation was performed during dry days' periods (Figure 17), in a slightly more frequent dynamics when compared with UNIT 1 and UNIT 2. Average water consumption for irrigation was 10 l/m^2. Garden area for Unit 3 is 30m^2 (6m x 5m). Total water requirements per irrigation process is thus 30 x 10l = 300 l. Distribution for the irrigation requirements is used in the simulation as follows:

[0,0,0,0,0,0,0,0,0,-300,0,-300,0,0,0,0,-300,0,-300,0,0,0,-300,0,0,0,0,0,0,0]

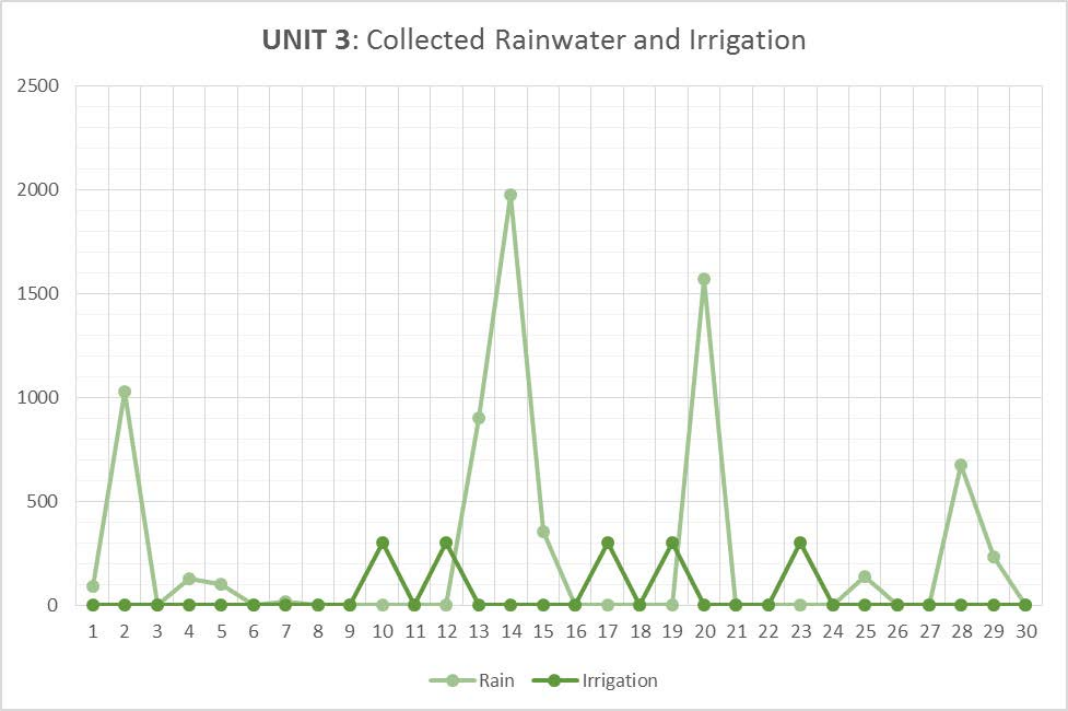

Figure 17. Irrigating the garden at UNIT 3 in zero-precipitation periods

3. Other, pseudo-random distribution based on occasional and unplanned water needs. The values were derived from the case-study observations and informal interviews with the consultants, and are approximated as follows:

[-7,-2,0,-1,0,-23,0,-2,-5,-1,0,0,0,0,0,-36,-2,-21,0,0,0,0,0,0,-6,-2,0,-1,0,0]

6.1.4 Scenario 1: Simulation Results and Discussion

An initial static resource allocation was performed by using simple mathematical formulas in spreadsheets, without utilizing self-sustainable mechanisms such as economy mode activation, delay requests, resource negotiation, etc.
An analysis of such static resource allocation without using the SSSHS infrastructure (Tables 8-10 and Figure 18 - Figure 21) presents the following relevant information:

- Unit 1 ceases to be self-sustainable at time unit 7
- Unit 2 ceases to be self-sustainable at time unit 11
- Unit 3 ceases to be self-sustainable at time unit 12
- All three units depleted their resources at time unit 12. This is consistent with information provided on real world case by the consultants.
- Due to inadequate storage facilities, a considerable amount of resources is lost on both Units 2 and 3. Specifically, resource loss occurred due to storages' limited capacities (1440l for Unit 2 and 1500l for Unit 3).
- From the time unit 14 and further, Units 2 and 3 are losing considerable overflow of resources [Figure 19, Figure 20], and in the same time units, Unit 1 is constantly depleted of resources.

The analysis shows both an inadequate storage infrastructure, and the need for advance resource allocation within the settlement. The case is so far consistent with the claim that a self-

sustainability can hardly be achieved by a single household(s) [122]. In this specific case, self-sustainability is not achieved, which is consisted with the real-world data.

Table 8. Daily production/consumption dynamics for Unit 1 without using SSSHS mechanisms

UNIT 1 *capacity: 2000 l*							
DAY	*Initial*	*Rain*	*Manual*	*L/P/D*	*Irrigation*	*Random*	*TOTAL*
1	700	76,5	60	-300	0	-2	534,5
2		855	0	-300	0	0	1089,5
3		0	40	-300	0	0	829,5
4		103,5	0	-300	0	-10	623
5		85,5	0	-300	0	0	408,5
6		0	40	-300	0	0	148,5
7		13,5	60	-300	0	-4	-82
8		0	60	-300	0	-1	-323
9		0	60	-300	0	0	-563
10		0	60	-300	-500	-12	-1315
11		0	60	-300	0	0	-1555
12		0	60	-300	-500	-1	-2296
13		751,5	0	-300	0	-1	-1845,5
14		1647	0	-300	0	0	-498,5
15		292,5	0	-300	0	-3	-509
16		0	60	-300	0	0	-749
17		0	60	-300	-500	0	-1489
18		0	60	-300	0	-21	-1750
19		0	60	-300	0	0	-1990
20		1309,5	0	-300	0	-1	-981,5
21		0	60	-300	0	0	-1221,5
22		0	40	-300	-500	-5	-1986,5
23		0	20	-300	0	-2	-2268,5
24		0	0	-300	0	-3	-2571,5
25		112,5	60	-300	0	0	-2699
26		0	60	-300	0	-7	-2946
27		0	0	-300	0	0	-3246
28		562,5	60	-300	0	-2	-2925,5
29		193,5	60	-300	0	0	-2972
30		0	60	-300	0	-1	-3213

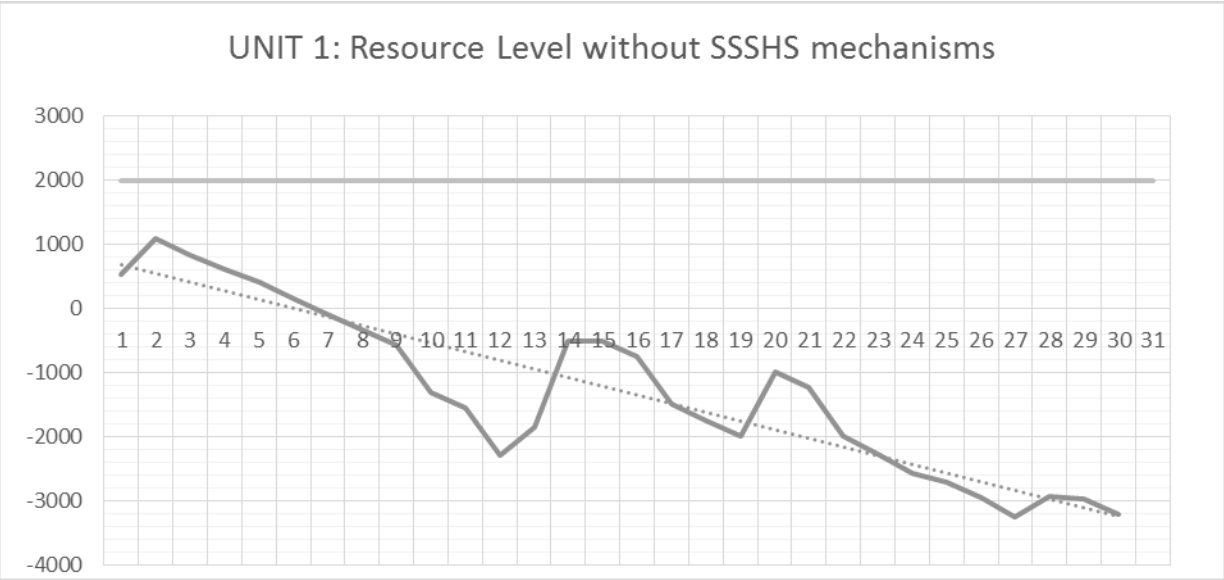

Figure 18. Resource level values without SSSHS mechanisms at Unit 1

Table 9. Daily production/consumption dynamics for Unit 2 without using SSSHS mechanisms

UNIT 2 *capacity: 1440 l*							
DAY	*Initial*	*Rain*	*Manual*	*L/P/D*	*Irrigation*	*Random*	*TOTAL*
1	350	63,75	0	-100	0	0	313,75
2		712,5	0	-100	0	0	926,25
3		0	40	-100	0	-25	841,25
4		86,25	0	-100	0	-70	757,5
5		71,25	0	-100	0	0	728,75
6		0	40	-100	0	0	668,75
7		11,25	20	-100	0	0	600
8		0	0	-100	0	-8	492
9		0	0	-100	0	0	392
10		0	20	-100	0	-32	280
11		0	40	-100	-300	0	-80
12		0	0	-100	0	0	-180
13		626,25	0	-100	0	-14	332,25
14		1372,5	20	-100	0	0	1624,75
15		243,75	40	-100	0	-17	1791,5
16		0	40	-100	0	0	1731,5
17		0	40	-100	0	0	1671,5
18		0	0	-100	-300	0	1271,5

19		0	0	-100	0	0	1171,5
20		1091,25	0	-100	0	0	2162,75
21		0	20	-100	0	-2	2080,75
22		0	0	-100	0	0	1980,75
23		0	40	-100	-300	-28	1592,75
24		0	0	-100	0	0	1492,75
25		93,75	40	-100	0	0	1526,5
26		0	20	-100	0	0	1446,5
27		0	0	-100	0	0	1346,5
28		468,75	20	-100	0	-5	1730,25
29		161,25	20	-100	0	0	1811,5
30		0	0	-100	0	0	1711,5

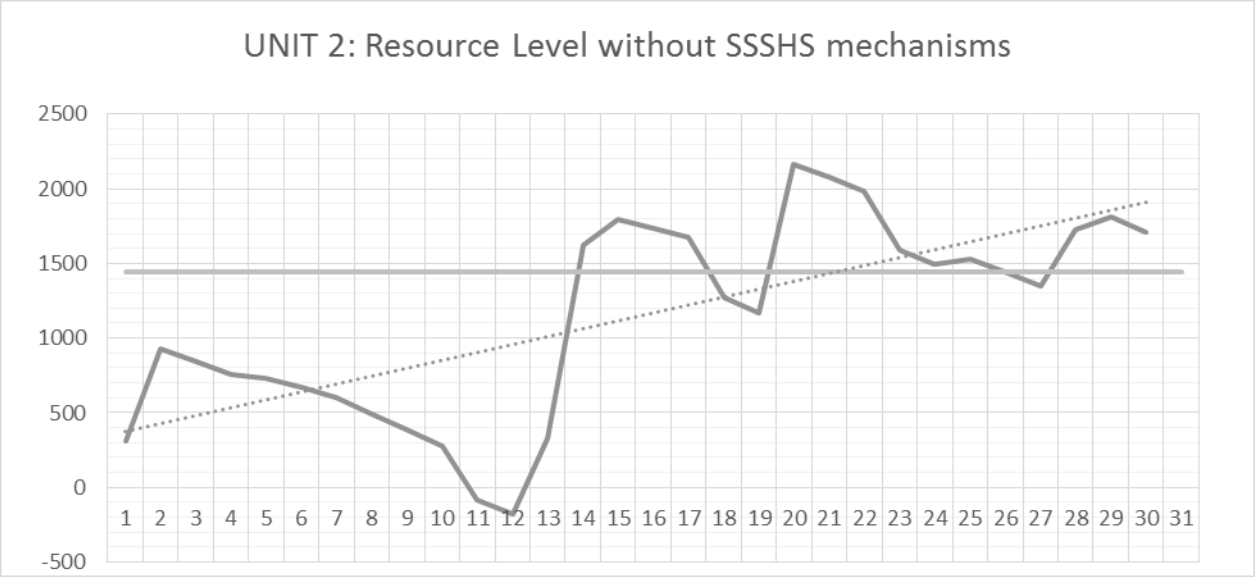

Figure 19. Resource level values without using SSSHS mechanisms at Unit 2

Table 10. Daily production/consumption dynamics for Unit 3 without using SSSHS mechanisms

UNIT 3 *capacity: 1500 l*							
DAY	*Initial*	*Rain*	*Manual*	*L/P/D*	*Irrigation*	*Random*	*TOTAL*
1	400	91,8	0	-100	0	-7	384,8
2		1026	0	-100	0	-2	1308,8
3		0	20	-100	0	0	1228,8
4		124,2	0	-100	0	-1	1252
5		102,6	0	-100	0	0	1254,6
6		0	0	-100	0	-23	1131,6
7		16,2	20	-100	0	0	1067,8
8		0	0	-100	0	-2	965,8
9		0	0	-100	0	-5	860,8
10		0	0	-100	-300	-1	459,8
11		0	20	-100	0	0	379,8
12		0	0	-100	-300	0	-20,2
13		901,8	0	-100	0	0	781,6
14		1976,4	0	-100	0	0	2658
15		351	0	-100	0	0	2909
16		0	0	-100	0	-36	2773
17		0	0	-100	-300	-2	2371
18		0	0	-100	0	-21	2250
19		0	40	-100	-300	0	1890
20		1571,4	0	-100	0	0	3361,4
21		0	0	-100	0	0	3261,4
22		0	40	-100	0	0	3201,4
23		0	0	-100	-300	0	2801,4
24		0	0	-100	0	0	2701,4
25		135	0	-100	0	-6	2730,4
26		0	0	-100	0	-2	2628,4
27		0	0	-100	0	0	2528,4
28		675	20	-100	0	-1	3122,4
29		232,2	20	-100	0	0	3274,6
30		0	0	-100	0	0	3174,6

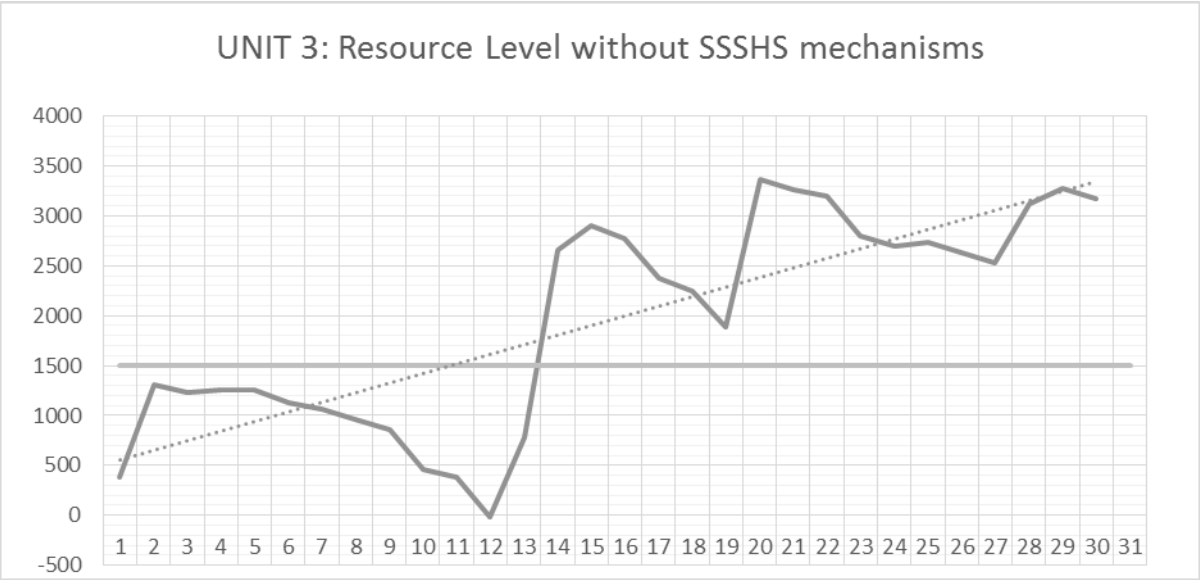

Figure 20. Resource level values without using SSSHS mechanisms at Unit 3

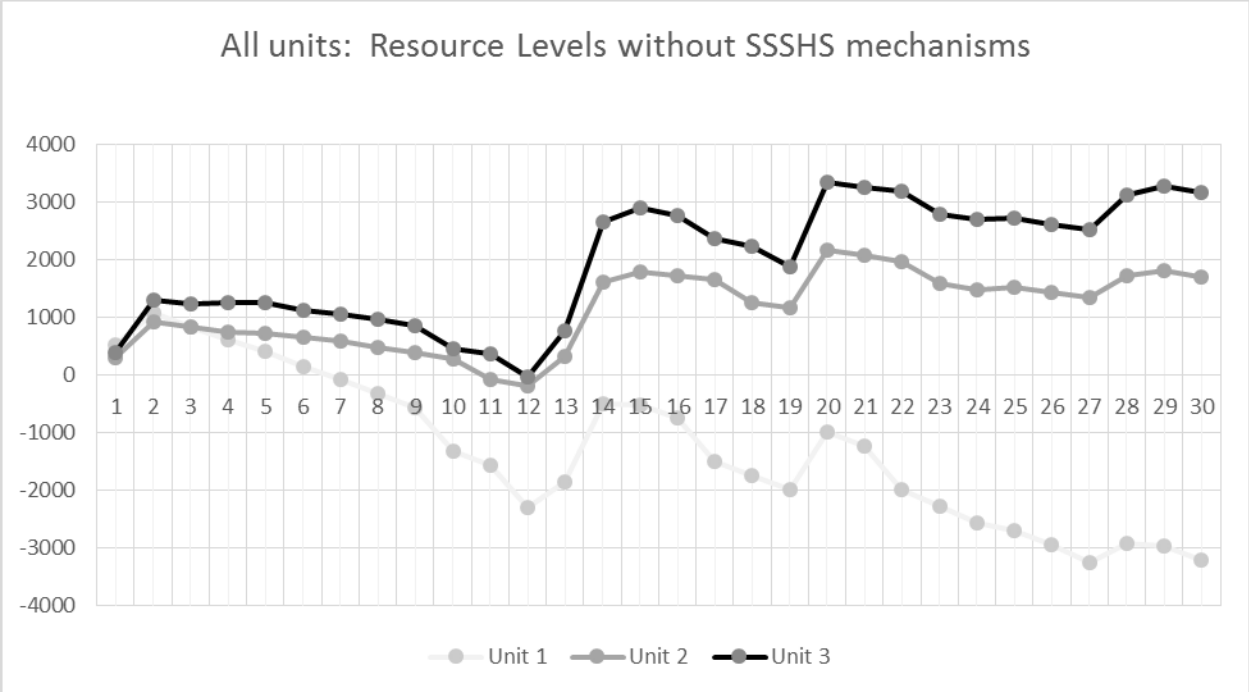

Figure 21. Resource level values without using SSSHS mechanisms for all three dwelling units

The simulation in the SSSHS framework was run on the same set of data used in the static resource allocation analyzed through spreadsheets. In the following simulation run all the implemented self-sustainability mechanisms were active. The complete listing of the implementation of scenario 1 model is available in Appendix B.

Resource level dynamics created by the SSSHS framework simulation run are presented separately (Figure 22 - Figure 24) and collectively (Figure 25) for all three dwelling units within the settlement.

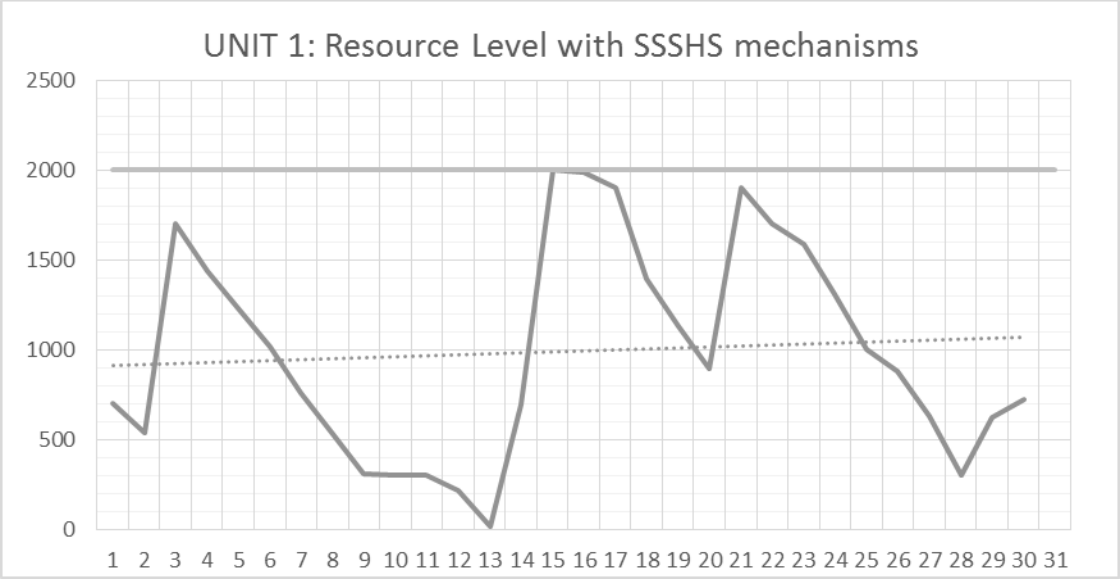

Figure 22. Resource level values by using SSSHS mechanisms at Unit 1

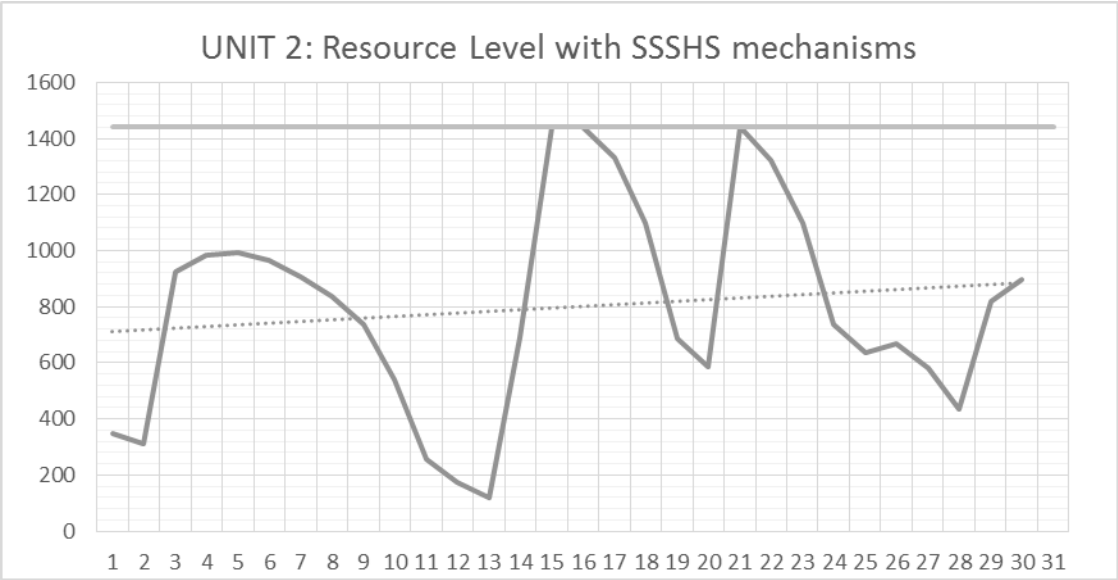

Figure 23. Resource level values by using SSSHS mechanisms at Unit 2

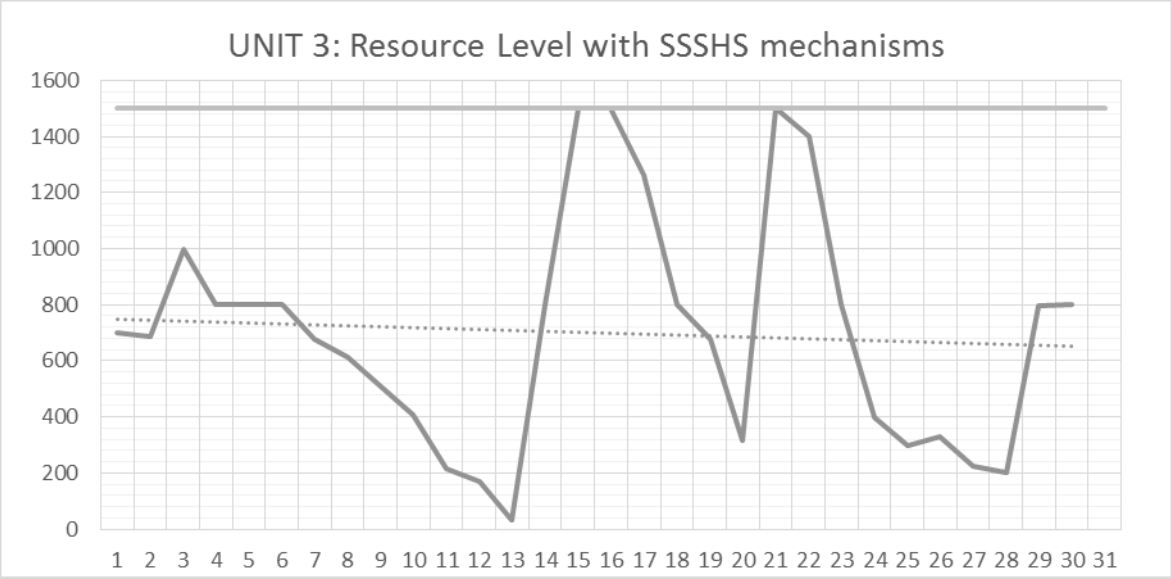

Figure 24. Resource level values by using SSSHS mechanisms at Unit 3

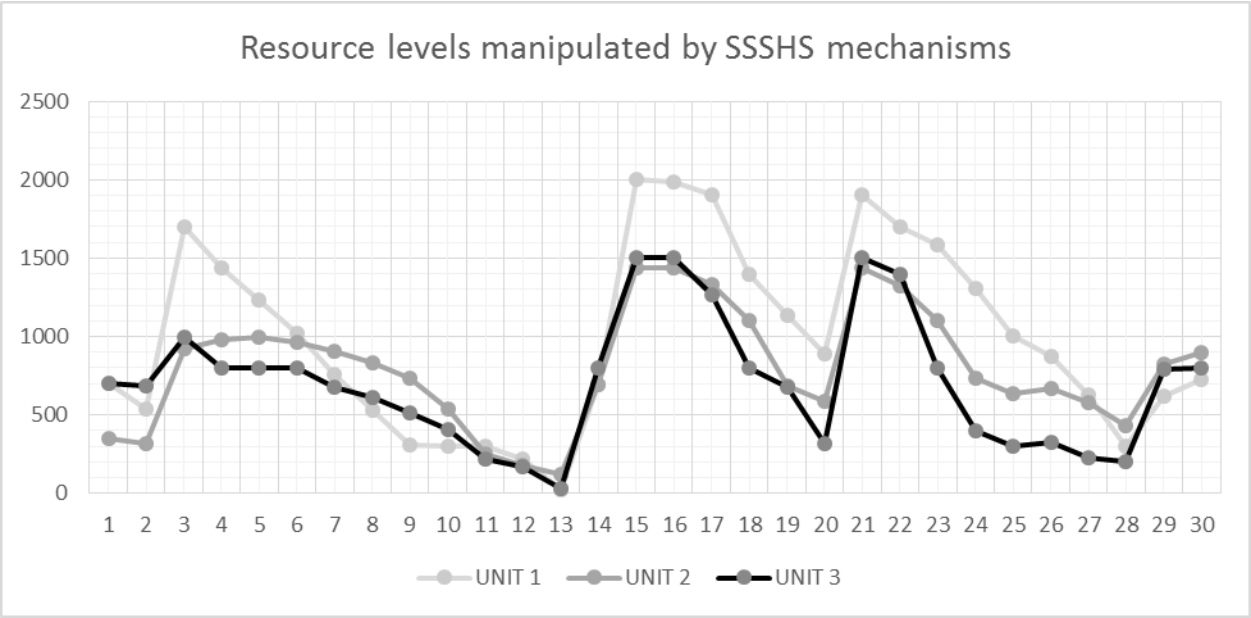

Figure 25. Resource level values by using the SSSHS mechanisms for all three dwelling units

From the simulation data visualization, it is evident that by using the SSSHS framework, the resource values are more balanced, less sloped, and the corresponding values in the same time units have considerably less offset. By using the SSSHS framework, the settlement would reach self-sustainability in the simulated time-period. Additional simulation data is given hereafter for further analysis:

```
.... [ END OF SIMULATION ] ....

******* Number of system interventions: 108
********** First intervention happened at time: 2

******* Number of LT ALERTS: 11

********** Number of DELAY requests: 11
********** Number of ECONOMY requests: 22
********** Number of NEGOTIATION requests: 11
```

******* Number of UT ALERTS: 24

********** Number of RESTORE requests: 16
********** Number of ADVANCE requests: 0
********** Number of GIVE requests: 48
********** Overflow of resources: 2804.850000

INDIVIDUAL REPORT FOR STORAGE UNIT-1

- CRL: 482
- UT alerts: 4
- Resources lost: 180.500000
- LT alerts: 5
- Economy reqs: 11
- Delay reqs: 0

CRL HISTORY: [700, 534.5, 1699.0, 1439.0, 1232.5, 1018.0, 758.0, 527.5, 307.4, 301.0, 301.0, 218.40000000000006, 18.30000000000006, 693.5, 2000, 1989.5, 1901.0, 1393.5, 1132.5, 892.5, 1901.0, 1699.0, 1589.0, 1307.0, 1004.0, 876.5, 629.5, 301.0, 621.5, 723.2]
OVERFLOW per time unit: [180.5]

INDIVIDUAL REPORT FOR STORAGE UNIT-2

- CRL: 796
- UT alerts: 7
- Resources lost: 908.550000
- LT alerts: 2
- Economy reqs: 5
- Delay reqs: 0

CRL HISTORY: [350, 312.75, 924.25, 982.55, 992.0, 964.85, 904.85, 836.1, 736.1, 541.5, 254.3, 176.0, 120.9, 692.05, 1440, 1440, 1329.5, 1099.0, 687.0, 587.0, 1440, 1321.0, 1099.0, 737.0, 634.0, 667.75, 580.75, 434.25, 821.0, 897.25]
OVERFLOW per time unit: [589.55, 181.75, 137.25]

INDIVIDUAL REPORT FOR STORAGE UNIT-3

- CRL: 699
- UT alerts: 13
- Resources lost: 1715.800000
- LT alerts: 4
- Economy reqs: 6
- Delay reqs: 0

OVERFLOW per time unit: [1175.4, 251, 289.4000000000001]

```
CRL HISTORY: [700, 684.8, 999.3, 799.0, 799.0, 799.0, 676.0, 612.2, 510.20000000000005,
405.20000000000005, 215.10000000000005, 171.0, 31.0, 799.0, 1500, 1500, 1263.0, 799.0,
678.0, 318.0, 1500, 1400.0, 799.0, 399.0, 299.0, 328.0, 226.0, 201.0, 795.0, 799.0]
```

Listing 7. SSSHS simulation report for the 1st scenario

The following three figures show resource levels at all three units to depict the difference in system dynamics when using the SSSHS mechanisms in contrast to without using them. The legend refers to the usage of the SSSHS mechanisms with the tag “ssshs”, and without the usage of the SSSHS mechanisms with the tag “wo” (abbreviation of “without”).

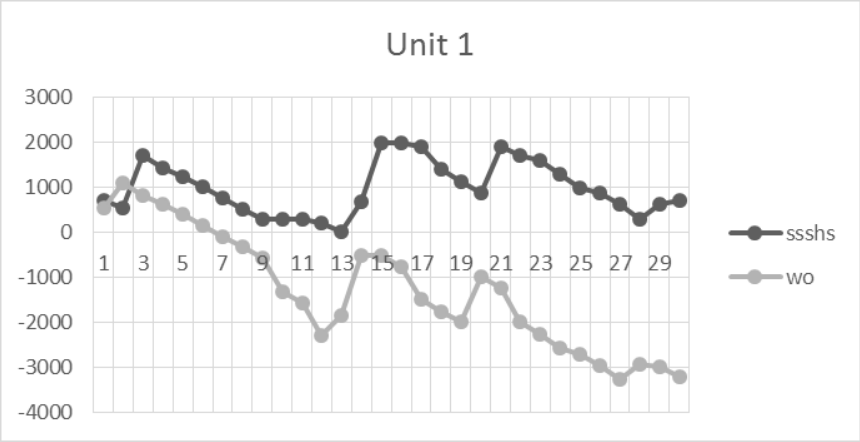

Figure 26. Resource levels with and without using the SSSHS mechanisms at Unit 1

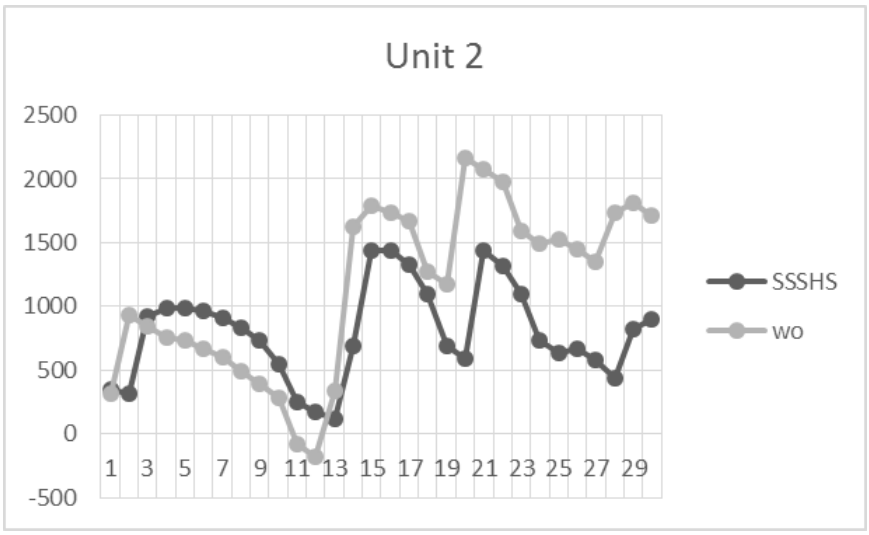

Figure 27. Resource levels with and without using the SSSHS mechanisms at Unit 2

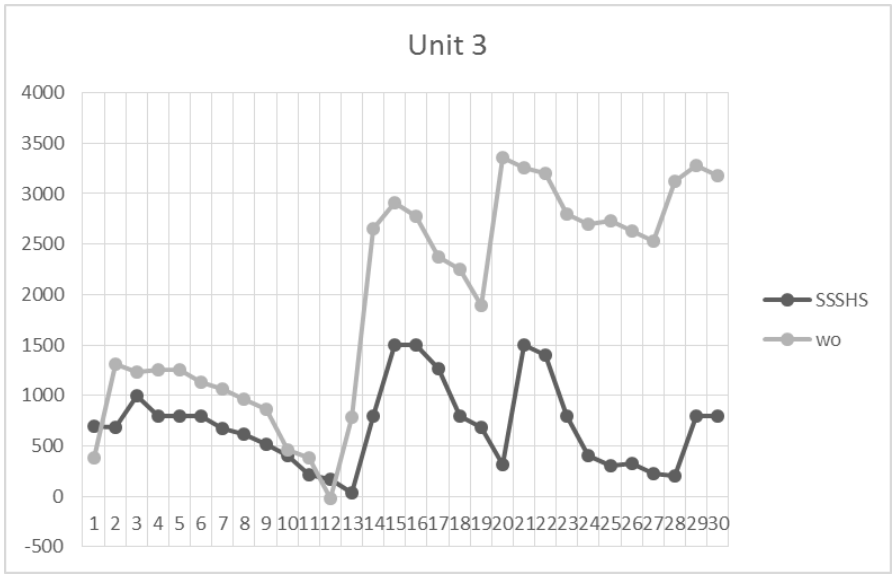

Figure 28. Resource levels with and without using the SSSHS mechanisms at Unit 3

Simulation ended with a total of 108 system interventions, which refer to any of the available self-sustainability mechanisms. This could indicate potential problems on the modeled system architecture which may be in a need for modification. First intervention happened early in the simulation run (time unit 2), and it was an upper threshold alert, meaning that there was a surplus of resources generated at Unit 3, possibly leading to the overflow and loss of certain amount of resources.

There was a total of 24 upper threshold alerts in the simulation run, and 11 lower threshold alerts. This could indicate that a model has more issues with the overflow of resources at specific dwelling units, than it has with the deficit of resources. A major amount of upper threshold alert triggered at Unit 3, which is consistent with the values observed in a static analysis presented in Table 10. This clearly indicated a non-optimal storage solutions for Unit 3, and should be considered for appropriate modifications.

A total loss of resources that took place because of the inadequate storage solutions was reported as 2804.85 liters of water: 180.5 liters at Unit 1; 908.55 at Unit 2; and 1715.8 liters of water at Unit 3. A modeler of the system should decide on which values of the loss are tolerable, and which values are demanding attention. In this case, Units 2 and 3 are the logical candidates for the revision of their storage solutions.

Lower threshold alerts are triggered five times at Unit 1, four times at Unit 3 and two times at Unit 2. Considering the fact that Unit 1 has largest set of consumers, these result were expectable.

All of the SSSHS self-sustainability mechanisms were requested for during the simulation run, and most of them have met the needed parameters for successful execution. Considering the lower threshold mechanisms, there was a total of 11 delay requests, 22 economy requests and 11 negotiation request. This shows a complex interactivity both inside the settlement dwelling units, and between them. Again, the upper threshold mechanisms were executed 64 times; 16 times the restore for the default capacity consumption was executed, and 48 times there was a surplus of resources which was given by a dwelling unit to another dwelling unit, in order to prevent the loss of the resource overflow.

6.1.5 Scenario 1: Recommendations based on the simulation output

As discussed in the previous chapter, the most indicative issue with the current settlement setup is considering storage capabilities. Dwelling units 2 and 3 frequently exhibit inadequate amount of

storage capacities, which seem too low, resulting in a considerable amount of lost resources which could be otherwise used in time periods of reduced income of resources.

A more detailed analysis of the history of resource levels per dwelling units shows more insight into the dynamics of resource allocation and storages' internal states on both static and SSSHS-supported resource allocation.
A static resource allocation could suggest that storage capacities for Unit 1, 2 and 3 should be 1089.5 l, 3602.75 l and 4861.4 l, respectively, in order to avoid loss of resources, when taking into account storages' resource levels through simulated time units. This would total to the needed storage capacity of the settlement to 9553.65 l. A resource allocation supported by the SSSHS mechanisms shows that storage capacities for Unit 1, 2 and 3 should be 2180.5 l, 2029.55 l and 2675.4 l, respectively, in order to avoid the loss of resources in the simulated instance. The needed storage capacity of the settlement supported by the SSSHS infrastructure totals to 6885.45 l, which offers a considerably less storage capacity needed by the settlement. Although this analysis is performed statically and the further simulation runs with changed parameters as discussed above would show new dynamics within the system, it should point to the relevant directions needed for the optimization of the system in the tradeoff on increasing the storage capacities in respect to the costs and physical implementation limitations of such actions.

6.1.6 Scenario 1 and Research Hypothesis H1

Research hypothesis H1 is stated as "*The agent-based framework for modelling and simulation of smart resource management in human settlements will prolong the time interval of self-sustainability when used on the developed test-bed scenarios*."

The static analysis, as well as the real-world case, showed that the observed human settlement failed to preserve the self-sustainability with the resource of interest (water) near the beginning of the observed time period.

With the aid of the SSSHS framework's self-sustainability mechanisms, the time interval of self-sustainability of the settlement was prolonged until the end of the simulation run, e.g. until the end

of the observable period, with identical input data. Thus, we can confirm hypothesis H1, facilitated by the developed test-bed referring to the permaculture scenario.

6.2 Scenario 2: Energy Production

Scenario 2 was designed and developed under the supervision of the consultant Loren Amelang, an expert in self-sustainable systems (Appendix A), and with partial reference to the observations registered at the existing eco-village setup which includes three dwelling units analog to the setup presented in scenario 1. The initial model parameterization was done in the context of stand-alone (off-grid) system pre-sizing project by using the software application called "PVsyst", described as a "*PC software package for the study, sizing and data analysis of complete PV systems*" [229]. Based on the energy production scenario, two simulations were executed according to the specifications given in the rest of this section.

The initial parameters included specifications of consumers and their load values, computed nominal power value of solar panel array, global system properties (panel tilt, azimuth, geographical coordinates, monthly meteorological data such as global irradiation, diffuse, temperature, etc.), specifications of the battery configurations, power input, etc. The solar irradiance input was obtained by MeteoNorm 7.1 station [230] on geographical coordinates of 45° N, 15° E, and altitude 128 m above sea level, for month January.

Table 11. Monthly meteorological values at Lat. 45.8°N, long. 16.0°E, alt. 128m (source: [229])

	Global Irrad. kWh/m².day	Diffuse kWh/m².day	Temper. °C	Wind Vel. m/s
January	1.02	0.56	1.1	1.50
February	1.98	1.16	3.7	1.70
March	2.94	1.55	8.1	1.99
April	4.03	2.19	12.8	1.90
May	5.53	2.46	18.5	1.80
June	5.83	2.54	21.2	1.70
July	5.85	3.03	23.0	1.60
August	5.03	2.35	22.5	1.49
September	3.27	1.66	16.5	1.48
October	2.19	1.36	12.6	1.40
November	1.11	0.67	7.5	1.40
December	0.76	0.54	2.3	1.49
Year	**3.30**	**1.68**	**12.5**	**1.6**

Monthly meteorological values (global irradiation, diffuse, temperature and wind velocity) and solar paths were used by the PVsyst software to obtain relevant values for the simulation's setup.

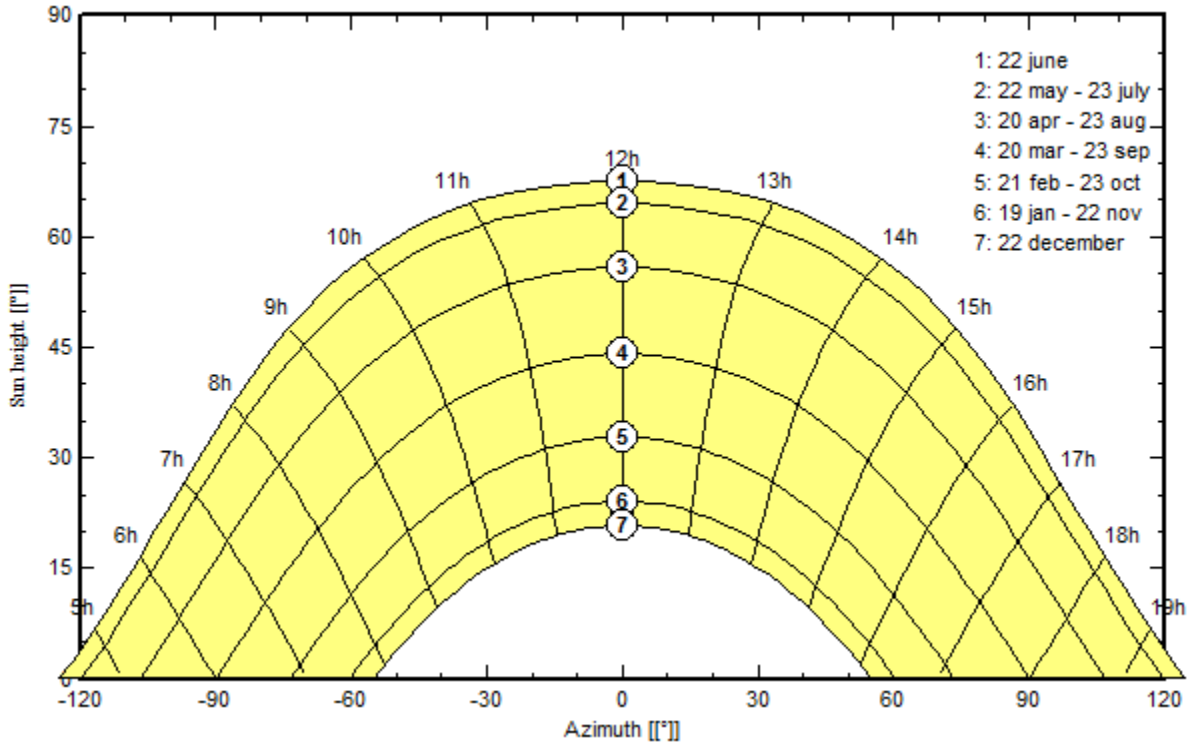

Figure 29. Solar paths at Lat. 45.8°N, long. 16.0°E, alt. 128m (source: [229])

Dwelling units are using photovoltaic solar panels as the main electrical energy source and propane generators as auxiliary energy source. The energy storage systems are implemented via solar battery packs. Units used in the scenario and in the SSSHS environment are **watt-hours** (Wh) for energy, **liters** (l) for propane quantities, **ampere hours** (Ah) for electric charge, and **watt-peak** (Wp) for the nominal power of the photovoltaic solar array (the term is used in colloquial language in the context of domestic photovoltaic installations, because the International System of Units prohibits the use of suffixes [231]). The term "**photovoltaic**" will be referred to with an abbreviation "PV" in most of the current section.

The initial system setup for all three units was designed and optimized with the help of PVsyst software for the planned and regular total load of the settlement units presented in the following tables. It is important to note that the cloth-washer at Unit 1 was added subsequently according to the scenario development described later in the section.

Table 12. Stationary consumers at Unit 1

UNIT 1				
	illumination	consumer electronics	fridge	cloth-washer
power (W)	20	65	*N/A*	*N/A*
number	3	3	*N/A*	*N/A*
approx daily use (h)	3	3	*N/A*	*N/A*
energy per cycle (Wh)	*N/A*	*N/A*	740	1100
daily energy (Wh)	180	585	740	1100
total (Wh)	**2605**			

Unit 2 uses hydronic pump, but does not utilize a cloth-washer during the observed time period.

Table 13. Stationary consumers at Unit 2

UNIT 2				
	illumination	consumer electronics	fridge	hydronic pump
power (W)	18	60	*N/A*	*20*
number	5	2	*N/A*	*1*
approx daily use (h)	3	2	*N/A*	*2,5*
energy per cycle (Wh)	*N/A*	*N/A*	740	*N/A*
daily energy (Wh)	270	240	740	50
total (Wh)	**1300**			

Unit 3 has the lowest initial load, thus the lowest requirements for solar photovoltaic array and solar battery pack.

Table 14. Stationary consumers at Unit 3

UNIT 3				
	illumination	consumer electronics	fridge	cloth-washer
power (W)	18	70	*N/A*	*N/A*
number	2	1	*N/A*	*N/A*
approx daily use (h)	2	2	*N/A*	*N/A*
energy per cycle (Wh)	*N/A*	*N/A*	0	0
daily energy (Wh)	72	140	0	0
total (Wh)	**212**			

The solar collector plane orientation was set with tilt of 45 degrees and azimuth of 0 degrees.

According to these values, the solar panel array nominal power for the Unit 1 was set to **1026 Wp**, and the battery pack capacity to **295 Ah**, which corresponds to **7080 Wh** with 24 V battery/system

voltage. Required autonomy was set to 4 days. More details on computation is available in the PVsyst help file [232]:

"*The evaluation of the available irradiance on the collector plane uses the Monthly Meteo tool algorithms, which calculate irradiation's monthly averages on the basis of instantaneous data for one day per month.*

This is not sufficient to manage the storage balance evolution from day to day, and the effective use of solar incident energy. Therefore the program generates a random sequence of 365 days, according to the algorithms of Collares-Pereira, renormalised to the monthly sums, and calculates the daily battery balance for three intervals in a day (morning, day and evening).

The accuracy is of the order of 10 - 20% (worst case for very tilted installations)."

The values generated by the PVsyst are presented in the following tables for all three units.

Table 15. Solar irradiance calculations for Unit 1

	Incid.	PV avail.	Demand	Excess	Missing
	kWh/m².day	*kWh*	*kWh*	*kWh*	*kWh*
Jan.	2.0	51.2	46.7	0.2	8.1
Feb.	3.1	71.5	42.1	24.5	0.0
Mar.	4.0	101.0	46.7	48.1	0.0
Apr.	4.5	109.8	45.2	59.3	0.0
May	5.4	136.5	46.7	84.2	0.0
June	5.3	129.9	45.2	79.3	0.0
July	5.4	137.5	46.7	85.3	0.0
Aug.	5.2	133.5	46.7	81.2	0.0
Sep.	4.1	100.6	45.2	50.1	0.0
Oct.	3.1	79.2	46.7	31.4	0.0
Nov.	1.9	48.0	45.2	1.2	6.7
Dec.	1.3	33.4	46.7	0.3	11.6
Year	**3.8**	**1132.2**	**549.3**	**545.1**	**26.4**

Unit 2 was allocated with the nominal solar array power of **849 Wp**, battery pack capacity of **255 Ah** at voltage of **24V (6120 Wh)**, with considering the required autonomy of 4 days.

Table 16. Solar irradiance calculations for Unit 2

	Incid.	PV avail.	Demand	Excess	Missing
	kWh/m².day	*kWh*	*kWh*	*kWh*	*kWh*
Jan.	2.0	42.4	40.3	0.0	4.7
Feb.	3.1	59.2	36.4	14.0	0.2
Mar.	4.0	83.7	40.3	38.4	0.0
Apr.	4.5	91.0	39.0	47.2	0.0
May	5.4	113.0	40.3	67.9	0.0
June	5.3	107.6	39.0	63.9	0.0
July	5.4	113.9	40.3	68.7	0.0
Aug.	5.2	110.5	40.3	65.4	0.0
Sep.	4.1	83.3	39.0	39.6	0.0
Oct.	3.1	65.5	40.3	21.4	0.0
Nov.	1.9	39.7	39.0	2.6	1.5
Dec.	1.3	27.7	40.3	0.0	15.5
Year	**3.8**	**937.5**	**474.5**	**429.2**	**21.9**

Unit 3 has a nominal solar array power of **141 Wp**, battery pack capacity of **42 Ah**, which translates to **1008 Wh** by using the 24V system voltage, and is considered with the required autonomy of 4 days.

Table 17. Solar irradiance calculations for Unit 3

	Incid.	PV avail.	Demand	Excess	Missing
	kWh/m².day	*kWh*	*kWh*	*kWh*	*kWh*
Jan.	2.0	7.1	6.6	0.0	0.7
Feb.	3.1	9.8	5.9	2.5	0.0
Mar.	4.0	13.9	6.6	6.5	0.0
Apr.	4.5	15.1	6.4	8.0	0.0
May	5.4	18.8	6.6	11.4	0.0
June	5.3	17.9	6.4	10.9	0.0
July	5.4	18.9	6.6	11.4	0.0
Aug.	5.2	18.4	6.6	11.0	0.0
Sep.	4.1	13.8	6.4	6.7	0.0
Oct.	3.1	10.9	6.6	3.5	0.0
Nov.	1.9	6.6	6.4	1.1	0.6
Dec.	1.3	4.6	6.6	0.0	2.5
Year	3.8	155.8	77.4	73.2	3.7

The analysis of the static resource allocation is presented via Table 18, Table 19 and Table 20. For the purposes of a more thorough insight into the resource allocation processes, factors such as production overflow on the battery pack were purposefully omitted in the static allocation.

Table 18. An individual resource allocation analysis for Unit 1

DWELLING UNIT 1

Storage: 295 Ah (24V) = 7080 Wh; Initial 5700 Wh; Array nom. power: 1026 Wp; Req. autonomy: 4 days

	producers *(Wh)*		***consumers*** *(Wh)*						***total*** *(Wh)*			***battery pack level***		*Propane*
day	**PV**	**Aux**	**Illum.**	**Electr.**	**Fridge**	**Cloth Washer**	**Tools**	**Inverter**	**prod.**	**cons.**	**diff.**	*(Wh)*	*(Ah)*	*(l)*
1	2970	0	138	289	724	0	0	115,1	8670	1266,1	7403,9	7403,9	308,50	0
2	2565	0	83	187	722	0	256	124,8	2565	1372,8	1192,2	8596,1	358,17	0
3	2236	0	242	487	729	0	0	145,8	2236	1603,8	632,2	9228,3	384,51	0
4	1277	0	192	437	722	0	0	135,1	1277	1486,1	-209,1	9019,2	375,80	0
5	1149	0	141	586	708	0	0	143,5	1149	1578,5	-429,5	8589,7	357,90	0
6	1080	0	154	291	759	0	0	120,4	1080	1324,4	-244,4	8345,3	347,72	0
7	1578	0	126	470	716	0	185	149,7	1578	1646,7	-68,7	8276,6	344,86	0
8	1757	0	202	256	709	0	0	116,7	1757	1283,7	473,3	8749,9	364,58	0
9	2100	0	169	561	774	0	750	225,4	2100	2479,4	-379,4	8370,5	348,77	0
10	1513	0	261	856	734	0	620	247,1	1513	2718,1	-1205	7165,4	298,56	0
11	1208	0	231	735	795	0	0	176,1	1208	1937,1	-729,1	6436,3	268,18	0
12	1255	0	169	730	745	0	732	237,6	1255	2613,6	-1359	5077,7	211,57	0
13	1005	0	280	610	738	1060	1635	432,3	1005	4755,3	-3750	1327,4	55,31	0
14	1001	0	113	575	737	0	0	142,5	1001	1567,5	-566,5	760,9	31,70	0
15	2841	0	247	494	739	0	326	180,6	2841	1986,6	854,4	1615,3	67,30	0
16	2549	120	274	950	757	1080	756	381,7	2669	4198,7	-1530	85,6	3,57	0,1
17	2684	240	132	775	732	0	1030	266,9	2924	2935,9	-11,9	73,7	3,07	0,2
18	2605	0	220	432	783	0	415	185	2605	2035	570	643,7	26,82	0
19	2156	840	261	396	799	1100	650	320,6	2996	3526,6	-530,6	113,1	4,71	0,7
20	2780	0	267	418	732	0	0	141,7	2780	1558,7	1221,3	1334,4	55,60	0
21	504	0	137	575	756	0	0	146,8	504	1614,8	-1111	223,6	9,32	0
22	731	0	227	855	798	1040	0	292	731	3212	-2481	-2257,4	-94,06	0
23	573	0	197	307	740	0	0	124,4	573	1368,4	-795,4	-3052,8	-127,20	0
24	432	0	273	369	719	0	0	136,1	432	1497,1	-1065	-4117,9	-171,58	0
25	688	0	219	381	743	0	0	134,3	688	1477,3	-789,3	-4907,2	-204,47	0
26	1987	0	165	916	763	0	0	184,4	1987	2028,4	-41,4	-4948,6	-206,19	0
27	3291	0	117	877	752	0	0	174,6	3291	1920,6	1370,4	-3578,2	-149,09	0
28	3428	0	175	270	777	1060	0	228,2	3428	2510,2	917,8	-2660,4	-110,85	0
29	2888	0	278	437	779	0	0	149,4	2888	1643,4	1244,6	-1415,8	-58,99	0
30	900	0	134	356	755	0	0	124,5	900	1369,5	-469,5	-1885,3	-78,55	0

The three dwelling units were initialized with the battery levels 5700 Wh, 4200 Wh, and 680 Wh, respectively. These values were calculated into the day one in both the individual static analysis, and in the SSSHS simulation environment.

Table 19. An individual resource allocation analysis for Unit 2

DWELLING UNIT 2 *Storage: 255 Ah (24V) = 6120 Wh; Initial: 4200 Wh; Array nom. power: 849 Wp; Req. autonomy: 4 days*														
	producers *(Wh)*		***consumers*** *(Wh)*						***total*** *(Wh)*			***battery pack level***		*Propane*
day	**PV**	**Aux**	**Illum.**	**Electr.**	**Fridge**	**Hydronic Pump**	**Tools**	**Inverter**	**prod.**	**cons.**	**diff.**	*(Wh)*	*(Ah)*	*(l)*
1	2191	0	268	232	775	63	0	133,8	6391	1471,8	4919,2	4919,2	204,97	0
2	2031	0	295	210	724	65	0	129,4	2031	1423,4	607,6	5526,8	230,28	0
3	1703	0	257	327	773	88	0	144,5	1703	1589,5	113,5	5640,3	235,01	0
4	1127	0	220	235	736	61	0	125,2	1127	1377,2	-250,2	5390,1	224,59	0
5	1267	0	276	212	772	44	0	130,4	1267	1434,4	-167,4	5222,7	217,61	0
6	1108	0	287	211	739	43	0	128	1108	1408	-300	4922,7	205,11	0
7	1694	0	248	350	731	86	0	141,5	1694	1556,5	137,5	5060,2	210,84	0
8	1811	0	284	248	736	50	365	168,3	1811	1851,3	-40,3	5019,9	209,16	0
9	2143	0	272	309	728	99	120	152,8	2143	1680,8	462,2	5482,1	228,42	0
10	2191	0	233	318	724	54	251	158	2191	1738	453	5935,1	247,30	0
11	1082	0	245	319	702	58	0	132,4	1082	1456,4	-374,4	5560,7	231,70	0
12	1270	0	243	209	734	51	348	158,5	1270	1743,5	-473,5	5087,2	211,97	0
13	1266	0	245	342	734	93	523	193,7	1266	2130,7	-864,7	4222,5	175,94	0
14	1192	0	227	318	731	63	0	133,9	1192	1472,9	-280,9	3941,6	164,23	0
15	1481	0	290	353	711	56	0	141	1481	1551	-70	3871,6	161,32	0
16	2135	0	245	357	779	47	0	142,8	2135	1570,8	564,2	4435,8	184,83	0
17	1368	0	272	344	752	50	0	141,8	1368	1559,8	-191,8	4244	176,83	0
18	2250	0	256	250	744	63	0	131,3	2250	1444,3	805,7	5049,7	210,40	0
19	2072	0	204	359	728	51	236	157,8	2072	1735,8	336,2	5385,9	224,41	0
20	1171	0	270	283	730	49	0	133,2	1171	1465,2	-294,2	5091,7	212,15	0
21	382	0	298	345	773	77	0	149,3	382	1642,3	-1260	3831,4	159,64	0
22	424	0	288	294	770	41	0	139,3	424	1532,3	-1108	2723,1	113,46	0
23	194	0	293	239	766	68	0	136,6	194	1502,6	-1309	1414,5	58,94	0
24	478	0	208	234	732	100	0	127,4	478	1401,4	-923,4	491,1	20,46	0
25	283	0	270	278	736	77	0	136,1	283	1497,1	-1214	-723	-30,13	0
26	1505	0	215	280	753	82	0	133	1505	1463	42	-681	-28,38	0
27	1722	0	237	260	753	65	0	131,5	1722	1446,5	275,5	-405,5	-16,90	0
28	2148	0	254	268	767	57	0	134,6	2148	1480,6	667,4	261,9	10,91	0
29	1725	0	232	321	772	82	0	140,7	1725	1547,7	177,3	439,2	18,30	0
30	996	0	244	247	719	83	0	129,3	996	1422,3	-426,3	12,9	0,54	0

It is evident that the Unit 3 has the lowest load requirements, but still occasionally uses auxiliary power generators to complement the insufficient energy production from the photovoltaic panels – in this case, a propane generator is utilized in time units 13, 14, and 23.

Table 20. An individual resource allocation analysis for Unit 3

DWELLING UNIT 3 *Storage: 42 Ah (24V) = 1008 Wh; Initial 680 Wh; Array nom. power: 141 Wp; Req. autonomy: 4 days*														
	producers *(Wh)*		***consumers*** *(Wh)*						***total*** *(Wh)*			***battery pack level***		*Propane*
day	**PV**	**Aux**	**Illum.**	**Electr.**	**Fridge**	**Cloth Washer**	**Tools**	**Inverter**	**prod.**	**cons.**	**diff.**	*(Wh)*	*(Ah)*	*(l)*
1	261	0	88	137	0	0	0	22,5	941	247,5	693,5	693,5	28,90	0
2	450	0	126	96	0	0	0	22,2	450	244,2	205,8	899,3	37,47	0
3	412	0	86	170	0	0	0	25,6	412	281,6	130,4	1029,7	42,90	0
4	64	0	96	92	0	0	0	18,8	64	206,8	-142,8	886,9	36,95	0
5	70	0	160	71	0	0	0	23,1	70	254,1	-184,1	702,8	29,28	0
6	41	0	156	197	0	0	0	35,3	41	388,3	-347,3	355,5	14,81	0
7	414	0	137	153	0	0	0	29	414	319	95	450,5	18,77	0
8	438	0	83	155	0	0	0	23,8	438	261,8	176,2	626,7	26,11	0
9	473	0	151	133	0	0	0	28,4	473	312,4	160,6	787,3	32,80	0
10	295	0	94	167	0	0	0	26,1	295	287,1	7,9	795,2	33,13	0
11	70	0	145	180	0	0	0	32,5	70	357,5	-287,5	507,7	21,15	0
12	124	0	126	105	0	0	0	23,1	124	254,1	-130,1	377,6	15,73	0
13	126	240	156	118	0	0	351	62,5	366	687,5	-321,5	56,1	2,34	0,2
14	75	240	86	169	0	0	0	25,5	315	280,5	34,5	90,6	3,77	0,2
15	370	0	157	139	0	0	0	29,6	370	325,6	44,4	135	5,62	0
16	321	0	99	89	0	0	0	18,8	321	206,8	114,2	249,2	10,38	0
17	344	0	87	138	0	0	0	22,5	344	247,5	96,5	345,7	14,40	0
18	402	0	119	79	0	0	0	19,8	402	217,8	184,2	529,9	22,08	0
19	326	0	134	124	0	0	0	25,8	326	283,8	42,2	572,1	23,84	0
20	307	0	138	72	0	0	0	21	307	231	76	648,1	27,00	0
21	21	0	141	200	0	0	0	34,1	21	375,1	-354,1	294	12,25	0
22	18	0	150	82	0	0	0	23,2	18	255,2	-237,2	56,8	2,37	0
23	35	120	98	136	0	0	0	23,4	155	257,4	-102,4	-45,6	-1,90	0,1
24	22	0	112	69	0	0	0	18,1	22	199,1	-177,1	-222,7	-9,28	0
25	23	0	101	141	0	0	0	24,2	23	266,2	-243,2	-465,9	-19,41	0
26	399	0	105	96	0	0	0	20,1	399	221,1	177,9	-288	-12,00	0
27	339	0	132	179	0	0	0	31,1	339	342,1	-3,1	-291,1	-12,13	0
28	414	0	102	124	0	0	0	22,6	414	248,6	165,4	-125,7	-5,24	0
29	362	0	127	85	0	0	0	21,2	362	233,2	128,8	3,1	0,13	0
30	240	0	88	91	0	0	0	17,9	240	196,9	43,1	46,2	1,92	0

The columns in these tables contain the following variables:

- *day*. A chronology of events is presented in days, which will be translated to simulation time units in the SSSHS framework.
- *PV*. A total amount of daily produced energy by the photovoltaic solar panel array system measured in Wh according to the observed time and location data.

- *Aux*. A total amount of daily produced energy by using the propane generator. Auxiliary fuel was used to prevent the depletion of energy reserves. Unit 1 was going to be depleted of the electrical energy in time unit 16, and used its total amount of 1 l of propane to postpone the depletion of energy reserves. Unit 3 was going to be depleted of the electrical energy in time unit 13, and used its total amount of 0.5 l of propane to postpone the depletion of energy reserves. Unit 2 lacks the support of auxiliary energy production systems. The propane generators were set according to the observed generic nominal and real values to produce 1.2 kWh of electrical energy by using 1 l of propane fuel.
- *Illum*. A total daily consumption of energy by using the domestic illumination.
- *Electr*. A total daily consumption of energy by using the consumer electronics like PCs, laptops, etc.
- *Fridge*. A total daily consumption of energy by the refrigerator units, according to manufacturer's specifications.
- *Cloth Washer*. A total daily consumption of energy by the cloth washer units, according to manufacturer's specifications.
- *Tools*. A total daily consumption of energy by using the power tools and machines like saws, drills, routers, trimmers, sanders, mixers, etc.
- *Inverter*. A total daily losses of energy due to the inefficiencies of the inverter DC/AC voltage conversion were approximated to 10% of overall AC consumption according to specifications of the inverter manufacturers.
- *prod*. A total daily production of energy, including the input from both PV panel arrays and propane generators.
- *consum*. A total daily consumption of energy.
- *diff*. A difference between total production and consumption values.
- *battery pack level*. Battery levels through time, measured both in Wh and Ah units. The upper battery capacities were purposefully omitted for the more in-depth analysis of energy dynamics.
- *propane*. A total daily amount of propane used in generators, in liters. Units 1 and 3 were having the limited reserves of propane, and Unit 2 does not utilize the propane generator.

The following two figures illustrate the production of electricity by using both the photovoltaic modules (“PV”) and auxiliary propane generators (“AUX”) in Units 1 and 3, against the battery pack energy levels (“battery”). All three variables are presented in Wh units.

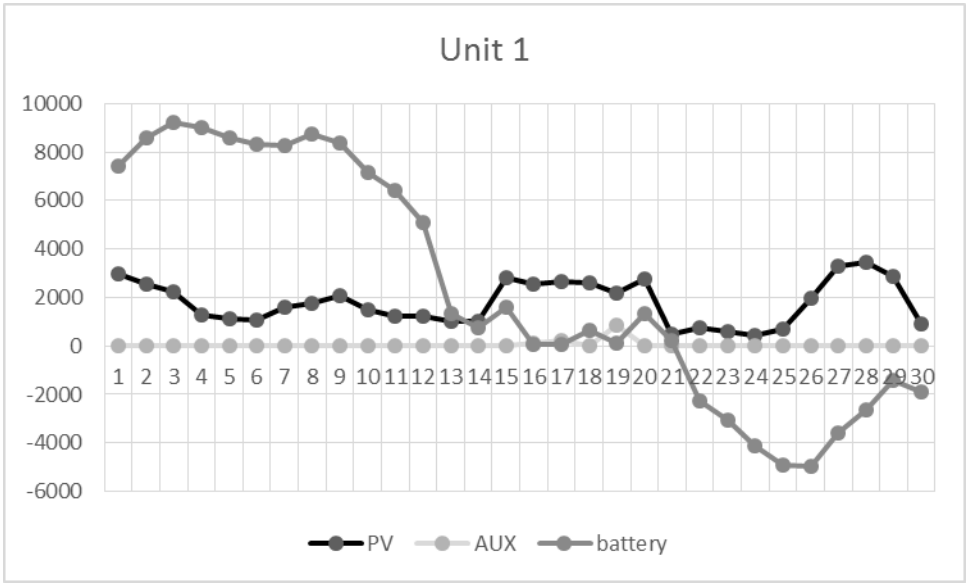

Figure 30. Electricity production by using PV modules and propane generators at Unit 1

Figures illustrate a significant deferment of the energy reserves depletion by the use of auxiliary propane generators. Unit 1 had the reserves of 1 l, and unit 3 reserves of 0.5 l of propane available for electricity production. Both units depleted their propane reserves in the developed scenarios.

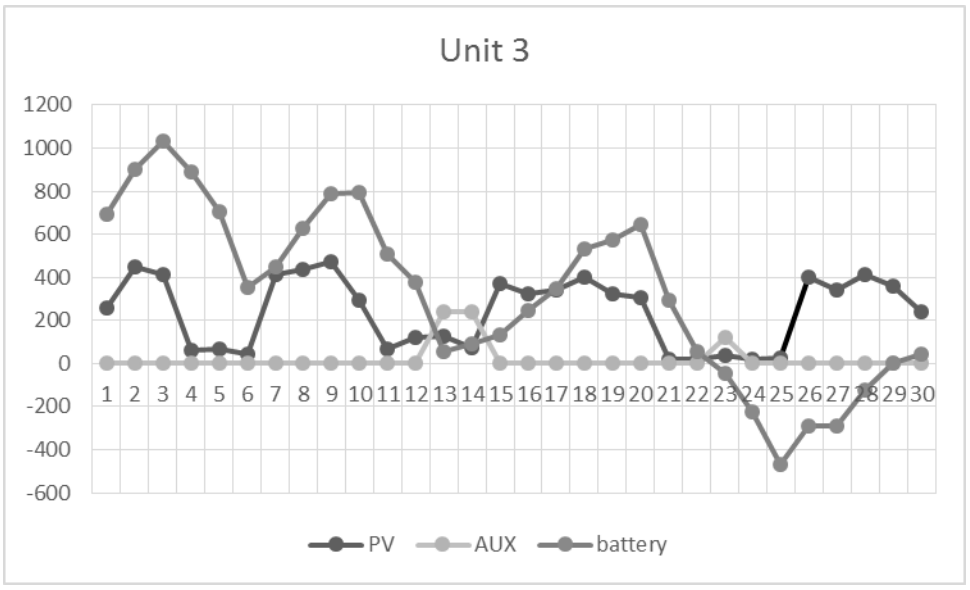

Figure 31. Electricity production by using PV modules and propane generators at Unit 3

The main event that rendered system specification at Unit 1 inefficient were unexpected difficulties with the house infrastructure that required immediate attention of the residents. Because of these new circumstances occurring on day 15, two more people moved in to Unit 1, and workings within and around the house intensified. The growing need for clothes washing, and a lack of time to do it manually, compelled the residents to obtain a washing machine, which presented a significant new load on the existing energy production infrastructure. In the similar way, refrigerator load, illumination, electronics', tools' and machines' consumptions increased. In order to mitigate the overall load on solar PV panel array, the residents used the propane generator for additional power production, but the energy reserves were completely depleted by the time unit 22.

Similarly, energy reserves were depleted at dwelling unit 2 at 25th day. The unit 3 managed to postpone the depletion of the energy reserves by using the propane fuel from day 13, but at the end depleted all the reserves at 23rd day.

6.2.1 Scenario 2, Simulation 1: PV array and auxiliary propane generators

The SSSHS framework was set to simulate **two different types of resources simultaneously**: the electrical energy obtained directly from the PV modules, and the propane fuel used to produce the electrical energy as an auxiliary source.

Within the SSSHS framework, the simulation runtime was set to 30 time units, which corresponds to days observed in the analysis of the individual static resource allocation scenario. All the parameters were initialized according to the observed static resource allocation data.

Results of the simulation are presented hereafter.

```
.... [ END OF SIMULATION ] ....

 ******* Number of system interventions: 163
 *********** First intervention happened at time: 1


 ******* Number of LT ALERTS: 26

 *********** Number of DELAY requests: 39
 *********** Number of ECONOMY requests: 38
 *********** Number of NEGOTIATION requests: 27


 ******* Number of UT ALERTS: 24

 *********** Number of RESTORE requests: 10
 *********** Number of ADVANCE requests: 0
 *********** Number of GIVE requests: 49
 *********** Overflow of resources: 1690.300000


INDIVIDUAL REPORT FOR STORAGE UNIT-1-PV
 - Capacity: 7080
 - CRL: 3561
 - UT alerts: 4
 - Advance reqs: 0
 - Resources lost: 1243.400000
 - LT alerts: 4
 - Economy reqs: 9
 - Delay reqs: 9
CRL   HISTORY:   [5700,   6499.0,   7080,   7080,   6870.9,   6441.4,   5929.8,   5861.1,
6175.100000000001,     4648.000000000001,     3450.800000000001,     2721.700000000001,
2006.5000000000011,     1110.900000000001,     1221.1000000000013,     2484.700000000001,
2203.7000000000007,     2963.8000000000006,     4168.500000000001,     4533.600000000001,
```

```
6097.4000000000015,    5186.500000000001,    3748.1000000000013,    3102.300000000001,
2181.1000000000013, 2001.0, 2001.0, 2001.0, 2155.4000000000015, 3742.6000000000013]
OVERFLOW per time unit: [611.1999999999998, 632.1999999999998]

INDIVIDUAL REPORT FOR STORAGE UNIT-1-Propane
 - Capacity: 20
 - CRL: 0
 - UT alerts: 0
 - Advance reqs: 0
 - Resources lost: 0.000000
 - LT alerts: 0
 - Economy reqs: 0
 - Delay reqs: 0
CRL HISTORY: [10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 9, 7, 7,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
OVERFLOW per time unit: []

INDIVIDUAL REPORT FOR STORAGE UNIT-2-PV
 - Capacity: 6120
 - CRL: 2800
 - UT alerts: 5
 - Advance reqs: 0
 - Resources lost: 306.200000
 - LT alerts: 11
 - Economy reqs: 16
 - Delay reqs: 16
CRL  HISTORY:  [4200,  5705.099999999999,  6120,  6120,  5869.8,  4435.400000000001,
4294.400000000001,  2737.9000000000005,  2574.2,  2501.0,  2784.5,  3396.6,  4102.2,
2833.7000000000003,  2359.9,  2501.0,  2501.0,  2501.0,  2765.1,  3496.2000000000007,
3724.500000000001,    4623.900000000001,    4999.0,    4667.4,    3647.9999999999995,
2265.700000000001, 1495.800000000001, 2394.700000000001, 2501.0, 2501.0]
OVERFLOW per time unit: [192.69999999999982, 113.5]


INDIVIDUAL REPORT FOR STORAGE UNIT-3-PV
 - Capacity: 1008
 - CRL: 799
 - UT alerts: 15
 - Advance reqs: 0
 - Resources lost: 140.700000
 - LT alerts: 11
 - Economy reqs: 13
 - Delay reqs: 14
CRL HISTORY: [680, 812.5, 1008, 1008, 865.2, 681.1, 601.0, 696.0, 799.0, 799.0, 799.0,
623.5, 601.0, 601.0, 720.6, 799.0, 799.0, 799.0, 799.0, 799.0, 799.0, 601.0, 601.0,
601.0, 601.0, 357.8, 535.6999999999999, 532.5999999999999, 697.9999999999999, 799.0]
OVERFLOW per time unit: [10.299999999999955, 130.4000000000001]

INDIVIDUAL REPORT FOR STORAGE UNIT-3-Propane
 - Capacity: 10
 - CRL: 0
 - UT alerts: 0
 - Advance reqs: 0
 - Resources lost: 0.000000
 - LT alerts: 0
```

```
 - Economy reqs: 0
 - Delay reqs: 0
CRL HISTORY: [5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0,
0, 0, 0, 0, 0]
OVERFLOW per time unit: []
```

Listing 8. Energy production simulation output (photovoltaic array + propane generators)

A total of five storage units reported at the end of the simulation run; three of them representing the **photovoltaic system** (“UNIT-X-PV”), and two of them representing the **auxiliary propane system** (“UNIT-X-Propane”).

According to the simulation output, a total of 163 system interventions executed during the simulation runtime, with similar number of triggering the upper (24) and lower (26) threshold alerts. Unit 3 had the lowest overall load of the three dwelling units, which reflected on its overall battery pack levels which showed above 50% capacity in most of the simulated time units. Units 1 and 2 were using below 50% of battery capacity about half of the simulation runtime, which should be checked against the used battery technology in the real-world implementation, where a possible destructive patterns of battery usage should be considered. The following three figures illustrate the differences between battery pack levels with and without using the SSSHS mechanisms.

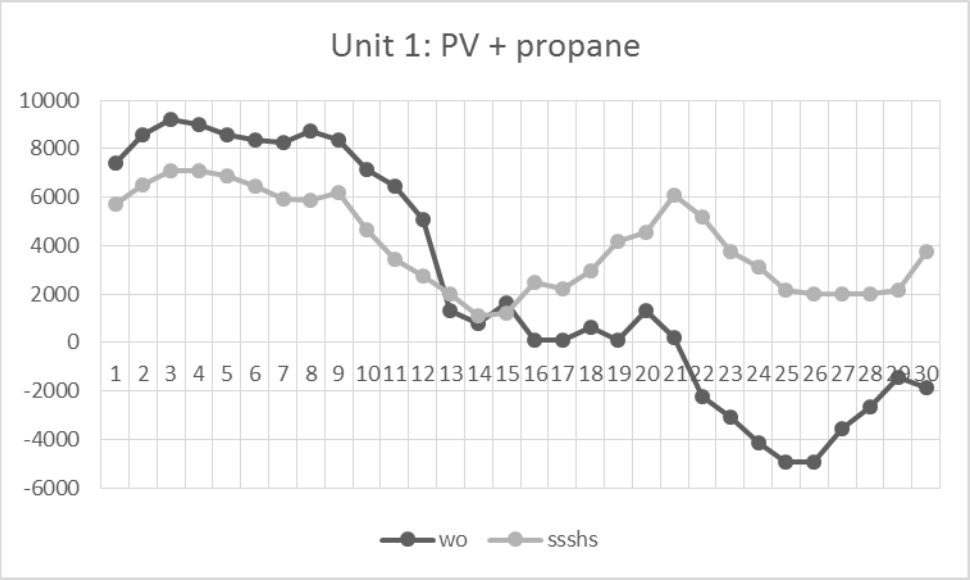

Figure 32. Battery pack levels with and without using the SSSHS framework at Unit 1

Figure 32 shows a decreased drop of battery levels with using the SSSHS mechanisms and maintaining the overall energy levels above zero until the end of the simulation runtime. Figure 33 demonstrates the dynamics of resource allocation at Unit 2, with the SSSHS mechanisms maintaining the energy levels of the battery pack until the end of the simulation runtime.

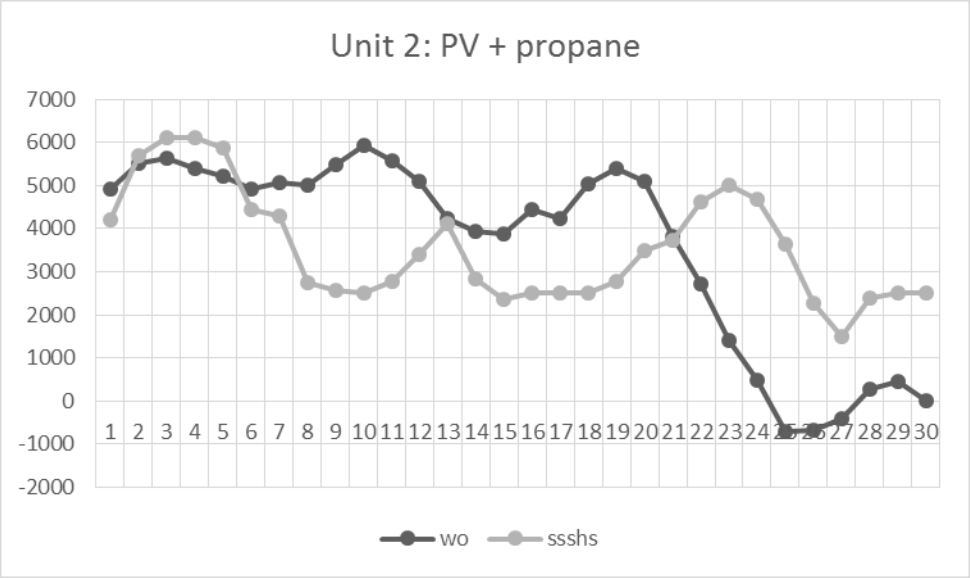

Figure 33. Battery pack levels with and without using the SSSHS framework at Unit 2

Figure 34 demonstrates the **additional ability of the framework to maintain more sensitive battery usage patterns**, significant for the long-termed battery health. These fluctuations can be additionally configured by the framework's parameters, particularly on the storage level.

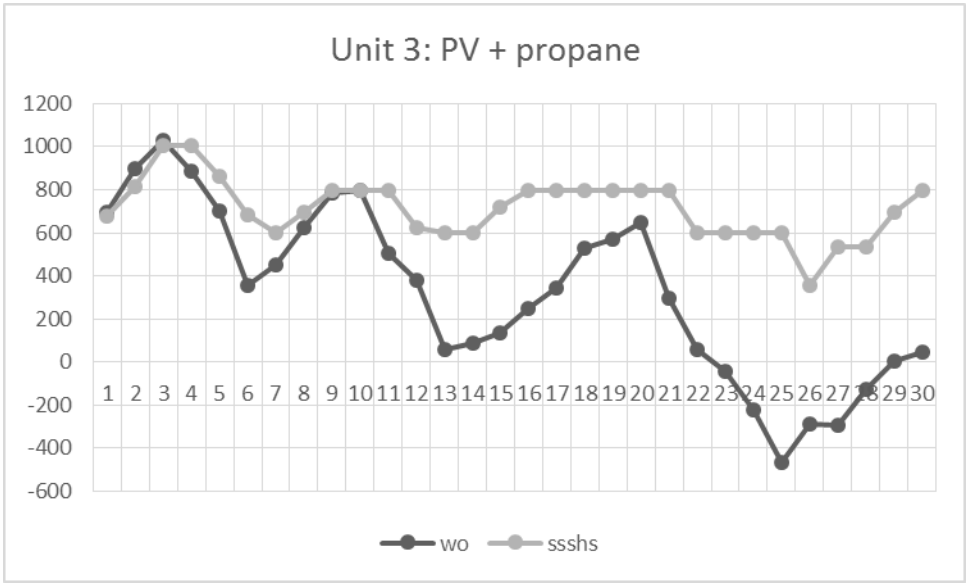

Figure 34. Battery pack levels with and without using the SSSHS framework at Unit 3

The analysis of the static resource allocation observed in worksheets showed that the settlement failed to preserve the self-sustainability of electrical energy production. The energy reserves were depleted in time units 22 for dwelling unit 1, time unit 25 for a dwelling unit 2 and in time unit 23 for the dwelling unit 3.

With the aid of the SSSHS framework’s self-sustainability mechanisms, the time interval of self-sustainability of the settlement was prolonged until the end of the simulation run, e.g. until the end of the observable period, with using identical input data. **Thus, we can confirm hypothesis H1, facilitated by the developed test-bed referring to this scenario**.

6.2.2 Scenario 2, Simulation 2: PV array without utilizing propane generator

The purpose of this simulation scenario was to observe if it would be possible to retain the self-sustainability of the energy production without using additional auxiliary generators, and keeping only the PV solar arrays as the only energy production source within the SSSHS simulation environment.

All the values were initialized according to the observed static resource allocation data, with the exception of auxiliary energy production by using the propane generators, which were excluded in this simulation run. The simulation runtime was set to 30 time units, which corresponds to days observed in the analysis of the static resource allocation scenario. The results of the simulation are presented hereafter.

```
.... [ END OF SIMULATION ] ....

 ******* Number of system interventions: 174
 ********** First intervention happened at time: 1

 ******* Number of LT ALERTS: 32

 ********** Number of DELAY requests: 45
 ********** Number of ECONOMY requests: 44
 ********** Number of NEGOTIATION requests: 34

 ******* Number of UT ALERTS: 20

 ********** Number of RESTORE requests: 10
 ********** Number of ADVANCE requests: 0
 ********** Number of GIVE requests: 41
 ********** Overflow of resources: 1690.300000

INDIVIDUAL REPORT FOR STORAGE UNIT-1-PV
 - Capacity: 7080
 - CRL: 2009
 - UT alerts: 4
 - Advance reqs: 0
 - Resources lost: 1243.400000
 - LT alerts: 7
 - Economy reqs: 12
 - Delay reqs: 12
CRL HISTORY: [5700, 6499.0, 7080, 7080, 6870.9, 6441.4, 5929.8, 5861.1, 6175.1,
4648.000000000001,     3450.800000000001,     2721.700000000001,     2006.5000000000011,
1110.900000000001,     981.1000000000008,     2174.200000000001,     1715.800000000001,
2266.3000000000006,    3471.000000000001,    2996.1000000000013,    4635.9000000000015,
3725.0000000000014, 2210.600000000001, 2001.0, 2001.0, 1511.7, 2000.8, 2001.0, 2001.0,
2234.4000000000005]
```

```
OVERFLOW per time unit: [611.1999999999998, 632.1999999999998]

INDIVIDUAL REPORT FOR STORAGE UNIT-2-PV
 - Capacity: 6120
 - CRL: 2800
 - UT alerts: 5
 - Advance reqs: 0
 - Resources lost: 306.200000
 - LT alerts: 12
 - Economy reqs: 17
 - Delay reqs: 17
CRL HISTORY: [4200, 5705.099999999999, 6120, 6120, 5869.8, 4435.400000000001,
4294.400000000001, 2737.9000000000005, 2574.2, 2501.0, 2784.5, 3396.6, 4102.2, 2593.7,
2359.9, 2501.0, 2472.7, 2501.0, 2765.1, 3496.2, 3648.5, 4547.9, 4999.0,
4111.200000000001, 2347.7000000000007, 1274.6000000000008, 15.600000000000819,
914.3000000000006, 1175.000000000001, 2501.0]
OVERFLOW per time unit: [192.69999999999982, 113.5]

INDIVIDUAL REPORT FOR STORAGE UNIT-3-PV
 - Capacity: 1008
 - CRL: 692
 - UT alerts: 11
 - Advance reqs: 0
 - Resources lost: 140.700000
 - LT alerts: 13
 - Economy reqs: 15
 - Delay reqs: 16
CRL HISTORY: [680, 812.5, 1008, 1008, 865.2, 681.1, 601.0, 696.0, 799.0, 799.0, 799.0,
623.5, 601.0, 601.0, 480.59999999999997, 629.4999999999999, 781.5000000000001, 799.0,
799.0, 799.0, 799.0, 601.0, 601.0, 601.0, 423.9, 180.7, 358.6, 355.5, 520.9,
649.6999999999999]
OVERFLOW per time unit: [10.299999999999955, 130.4000000000001]
```

Listing 9. Energy production simulation output (photovoltaic array only)

It is evident that there were more system interventions in the simulation scenario with fewer energy production capabilities, indicating an increased struggle of the framework to maintain the self-sustainability of energy production. The differences in energy reserves during the observed time frame, between using the SSSHS mechanisms and without using them, are illustrated in the following three figures.

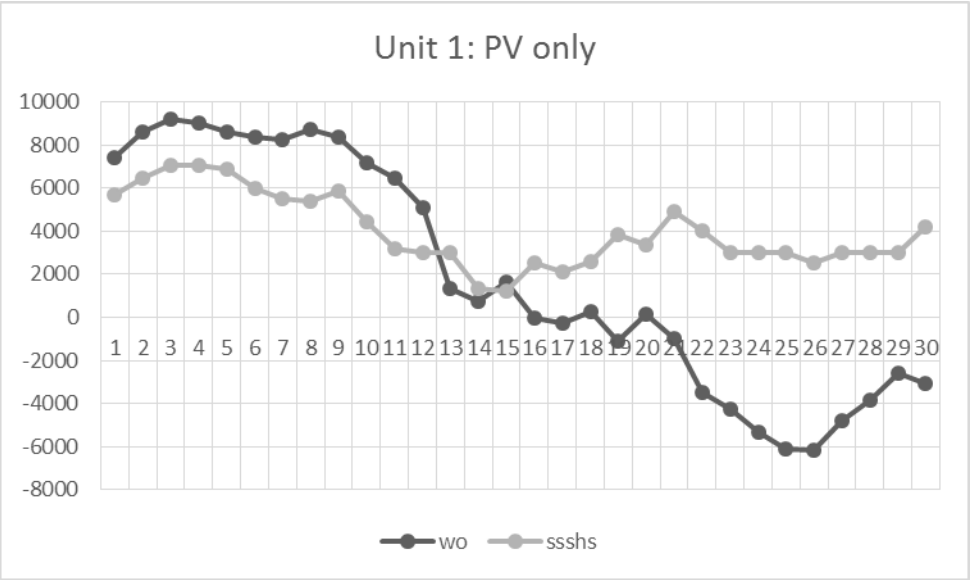

Figure 35. Battery pack levels with and without using the SSSHS framework at Unit 1

Similarly to the results generated in simulation 1 of energy production scenario, results of the carried simulation 2 presented via graphs also illustrate a more even battery usage patterns, which is especially apparent at Unit 3 via Figure 37. Additional experiments and simulations could provide a valuable insight into the possibilities of direct manipulation of the battery usage patterns, where several key parameters of the SSSHS framework could facilitate such manipulations.

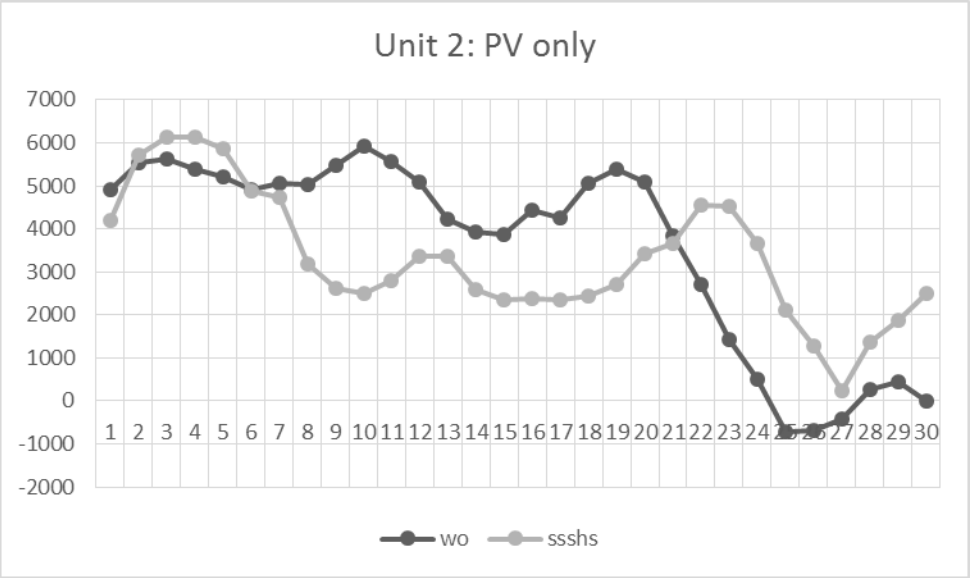

Figure 36. Battery pack levels with and without using the SSSHS framework at Unit 2

The graphs also indicate that by using the SSSHS framework, the energy levels are kept above zero until the end of the simulation run.

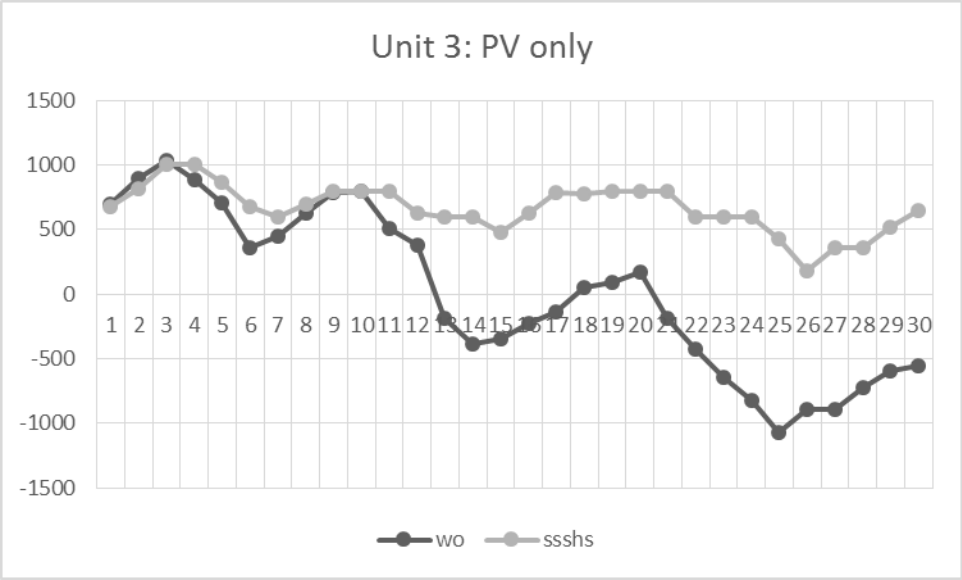

Figure 37. Battery pack levels with and without using the SSSHS framework at Unit 3

The analysis of the static resource allocation observed in worksheets with omitted propane generators showed that the settlement failed to preserve the self-sustainability of electrical energy production. Because in this scenario the propane generators, as auxiliary energy production units, were purposefully omitted from the analysis, the energy reserves were expectedly depleted sooner than in previous scenario where the generators were able to temporarily postpone the depletion; in this case, dwelling units 1, 2 and 3 were depleted of energy reserves in time units 16, 25, and 13, respectively.

The simulation results showed a key importance of lower and upper threshold settings in the simulation environment. Initial values for Unit 1 were set to 3000 Wh and 6000 Wh, respectively, which ended the simulation run at 26th time unit, prolonging the self-sustainability of the settlement significantly, but ultimately declaring the system as non-self-sustainable. By modifying the values to 2000 Wh and 6500 Wh, the framework was able to **prolong the self-sustainability of the system until the end of the simulation runtime**. These results are pragmatically curious in considering the physical implementation of the battery units and their usage, dealing with the complex

trade-off between a prolonged self-sustainability of the system versus optimal battery usage, which depends on the battery technology and manufacturer's recommendations; the prolonged battery usage with low capacity might prove destructive for the battery in the long run and the modeler should consider a number of the lower threshold alerts, along with the CRL variable values' analysis through simulation run, in designing the system with specific technologies. Another possible solution in dealing with prolonged battery usage beneath destructive thresholds which can be relatively easily implemented within SSSHS framework is to design such a storage unit which does not allow a battery usage beneath such thresholds, by appropriately modifying maximum capacity and lower thresholds of the storage unit which refers to the battery pack in question. Declaring, for example, 50% of actual battery pack capacity as the maximum storage capacity of the referring storage unit is the simplest of solutions, although not as flexible in terms of occasional short usages beneath the threshold.

6.2.3 Conclusions on the two simulations based on the 2nd scenario

Two resources were simulated simultaneously in the executed simulation 1, where the SSSHS framework was able to prolong the self-sustainability of the settlement regarding the electrical energy reserves stored via solar battery packs. Settlement units used photovoltaic solar modules and auxiliary propane generators for the purposes of producing the electrical energy needed for residential illumination, consumer electronics, domestic appliances, working tools, etc. Relevant losses of the energy were calculated appropriately (e.g. losses generated by the power inverter), and the scenario setup was validated by the consultant and the expert in the off-grid systems Loren Amelang.

Simulation 2 was executed for the purposes of testing the self-sustainability of the settlement in the identical initial setup, but without using the auxiliary generators. The results of the executed simulation run showed that the SSSHS framework's mechanisms were able to retain the self-sustainability of the observed settlement even without the auxiliary generators.

A static resource allocation analysis presented via spreadsheets showed that the loss of energy reserves in the observed settlement occurred in time units 22 for dwelling unit 1, time unit 25 for a

dwelling unit 2 and in time unit 23 for the dwelling unit 3, in the case of using the auxiliary generators. When omitting the auxiliary generators, dwelling units 1, 2 and 3 were depleted of energy reserves in time units 16, 25, and 13, respectively, thus significantly sooner.

In the SSSHS framework environment, both simulations showed that the SSSHS mechanisms were able to prolong the self-sustainability of the settlement until the end of the simulation runtime, thus confirming the hypothesis H1.

Additionally, the framework showed to be useful in analyzing, manipulating and designing the battery pack usage patterns, which can be modified according to the implemented technology, by changing the framework's parameters such as upper and lower thresholds, negotiation and inter-dwelling resource transfer parameters, etc.

6.3 Space Environments Scenarios

In contrast to the environments on Earth, "*space environments are characterized by isolation (a separation from the normal or daily physical and social environment), confinement (restriction within a highly limited and sharply demarcated physical and social environment), deprivation, and risk*". [216] While events such as depletion of resources, resulting in a habitats non-self-sustainability, may be tolerable in certain scenarios based on Earth (new resources could be transported, relocation of residents could be possible, and similar solutions may be derived), such event in a space environment could result in human fatalities.

There is a need „*to develop systems that produce food, purify their water supply, regenerate oxygen and remove undesirable components of air*". [217] Recycling processes facilitate critical input of resources in space environments. In the SSSHS framework, this input may be considered and implemented as a projected resource production distribution, deliberately devoid of the technical details of its implementation mechanisms.

As remarked in [218], space crafts, orbital space stations, off-Earth settlements, and similar isolated environments, are dependent upon the constant shuttling of resources to and from the Earth's surface, or are limited by the finite resources which they had carried at the outset. Such assumption would imply a rigid linear process in resources' dynamics, and could not facilitate self-sustainability as defined in the context of this research. Mechanisms such as recycling, and collecting available resources in situ, together with a smart resource management automation, could provide grounds for a long-term self-sustainability. Because within the space environment "*life proceeds without abundant provisions or supplies*" [219], smart resource management mechanisms in such scenarios could prove vital to habitat's operations.

The agents that a group of authors are developing at Essex [154], [219-220] have embedded the equivalents of Asimov's laws in the form of rules, slightly modified and implicit to the design of the behaviour arbitration mechanism [218]:

1. Protect the habitat (and as a consequence the occupants)

2. Obey authorized stakeholders (but commonly all building occupants)
3. Maximize comfort for individual occupants.
4. Economize energy

These include safety and emergency behaviours, explicit manual overrides over automation, learning behaviours, and rationality regarding resource consumption. In the SSSHS framework, protecting the habitat would be implicit to the agent's behaviours, by which they are maintaining resource quantities at safe levels at all times (as long as possible). Obeying authorized stakeholders could refer to the possibility of changing the parameters of the system at any time. Comfort and economization of the resource consumption are also implicitly embedded in the SSSHS framework, where a modeler can determine and set the parameters regarding user comfort within each consumer agent related to user comfort context.

A design study of space settlements carried by NASA [221] remarks that "*there must be enough water to sustain life and to maintain sanitation*", and suggests an adequate diet for a "reasonable environmental stress and a heavy workload", which consists of about 3000 Cal/day:

- 2000 g of water
- 470 g dry weight of various carbohydrates and fats
- 60 to 70 g dry weight of proteins
- adequate quantities of various minerals and vitamins

Interestingly, the study also suggests effective limitation to resource consumption, namely the use of "*recirculating showers, low volume lavatories, and efficient use of water in food preparation and waste disposal*". These notions would be considered in creating both the space colony and the prolonged space flight scenarios.

It is important to note that, although the relevant space-related scenarios' parameters were observed and derived from the literature presented at sections 6.2, 6.3 and 6.4, the scenarios use abstractions devoid of the myriad of technical details of the technology implementation mechanisms, physical space-related parameters and similar factors which might interfere with the self-sustainability. However, because the SSSHS framework works with final number values of such factors and

mechanisms to which the specific meanings are attached in the scenario implementation process, such factors can be additionally implemented into the framework scenario parameters by the experts on the fields of space travel, colonies, and similar fields of research.

6.3.1 Scenario 3, Simulation 1: Space Colony

As remarked in [218], should one of the possible catastrophes within the foreseeable future wipe out a significant percentage of the earth's current natural ecosystems and species, there would be a need for leaving Earth and colonize another planet. With the current technology and taking into regard that no other revolutionary space propulsion systems might be realistic and exploitable (FTL drives or similar), and regarding the natural (in)habitability of the reachable planets, the most likely solution that would be feasible relates to the design and development of the artificial, habitable environments, that would be placed or built on the target planet.

Water consumption dynamics (as well as other resource consumption and metabolic exchange rates with the habitat) within personnel is dependent on several key factors such as the duration and the effort of the crew's activities (Figure 38), however for the purpose of the SSSHS scenario, water use for residents is averaged to 50 l/day/person, as in the permaculture scenario: "*Since Mars has gravity, water can be used in the same way as on Earth. Each astronaut will be able to use about 50 liters of water per day. The water will be recycled, which takes much less energy than extracting it from the Martian soil. Only water that can not be recycled will be replaced by water extracted from the soil.*" [222]

Therefore, simulation uses an even consumption distribution per instantiated astronaut:

[-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50]

An independent assessment of the technical feasibility of the Mars One mission plan made by a group of authors from MIT [223] reveals some of the details regarding the mission, which will be reviewed and partially used in building the SSSHS space colony scenario. As revealed in the mission plan, the campaign will start with a precursor mission, which will be launched in 2018. The mission involves a Mars surface lander, which is based on the design of the NASA Phoenix Lander. The main goal of this precursor mission is to "*test and demonstrate a series of key technologies required to sustain a human settlement on the Martian surface.*" Technologies in question refer to thin-film solar arrays and, particularly curious for the SSSHS scenario, an oven to extract water from Martian regolith.

For the first 500 day period of the mission, a rover will be delivering regolith to the ISRU (In-Situ Resource Utilization) oven, where it will be baked to extract water. Because there is no feasible technology available for a continuous resource deployment from Earth to Mars, In-Situ Resource Utilization is considered as a fundamental factor for ensuring self-sustainability of the settlement. Consistent with the self-sustainability in the context of this research, The Mars One mission plan supports the philosophy of maximizing local resource use and exploiting of existing technology [224].

A portion of the extracted water will be electrolyzed in order to generate oxygen, and it is expected that, by the time that the first crew of 4 people departs from Earth, the ISRU system would have produced 3000 l of water in the case of not growing food locally, and 14000 l of water in the case of growing food locally.

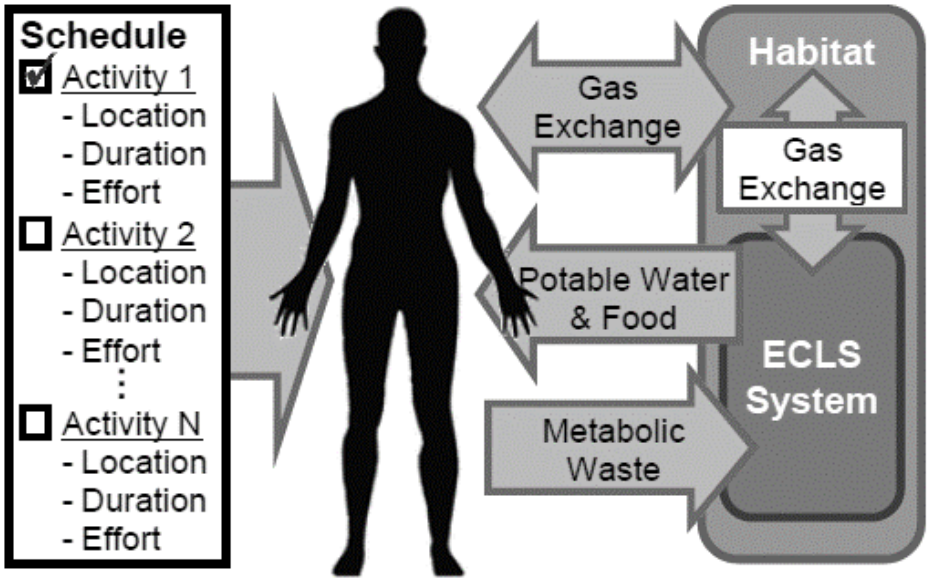

Figure 38. Data Flow within the Habitation Module for the Mars One mission plan [136]

In the SSSHS scenario, 14000 l will be used as an initial resource value distributed through the several habitation units' storages, as growing food locally in the context of this research is an important factor for long-term self-sustainability. Additional water production could be described

with further ISRU water extraction processes, as well as with water recycling from urine, greywater, evaporations, etc.

The International Space Station (ISS) Water Recovery and Management (WRM) System is designed to ensure the availability of "*potable water for crew drinking and hygiene, oxygen generation, urinal flush water, and payloads as required*." In order to facilitate this function, waste water is collected from the crew urine, humidity condensate, and Sabatier product water. This waste water is then and subsequently processed by the Water Recovery System (WRS) to potable water. [225]. The ISS Water Recovery and Management System is visualized in Fig 19.

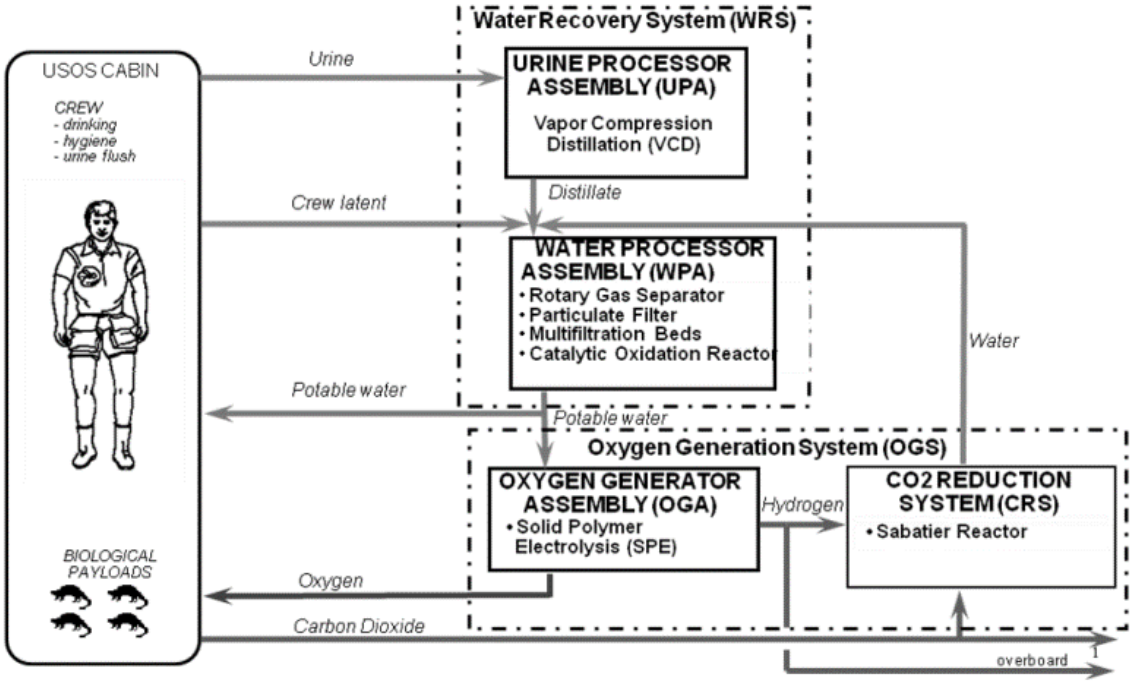

Figure 39. ISS Water Recovery and Management System [138]

Theoretical recovery rate of the Urine Processor Assembly (one of the core components of ISS WRS) is designed to be 85%. SSSHS scenario uses this information to create input resource distribution.

6.3.1.1 Scenario 3 Detailed

SSSHS space-colony scenario will include four initial habitation units, two of them reserved for the human crew activities, and two of them dedicated for crop cultivation. Because of the designed flexibility of the SSSHS framework, it is possible to later expand this number, enabling the infrastructure to grow along with its increasing population in the future.

The simulation duration for the SSSHS space colony scenario can be derived from time period between launch windows from Earth to Mars. This time period is the minimum continuous period over which the settlement must be self-sufficient, because there would be no opportunities for a resource resupply from Earth. It is set to 26 months, or 780 days, or 195 weeks. In order to create a feasible scenario, time units will be divided to half a month each, resulting in a 52 time units.

The analysis of the Technical Feasibility of the Mars One Mission Plan also showed that the 200m^2 of plants required up to 150 l of water per hour, or 3600 l per day. In the case of growing the food locally, it is estimated that the initial water volume needs to be set on 14000 l.

The initial setup of the four units is as follows.

Unit 1: human habitation unit

Unit 1 is assigned for human habitation, and will be initially populated by the minimal crew of two people. The remaining unit setup is planned as follows:

- 1 water storage unit with maximum capacity 5000l
- Initial resource volume: 2000l
- Two resource inputs: WRS and ISRU analogies, working constantly throughout the duration of the mission

Unit 2: human habitation unit

Unit 2 is also assigned for human habitation, and will be initially equally populated by the minimal crew of two people. The remaining unit setup is planned as follows:

- 1 water storage unit with maximum capacity 5000l
- Initial resource volume: 2000l

- Two resource inputs: WRS and ISRU analogies, working constantly throughout the duration of the mission

Units 3 and 4: crop cultivation units

Units 3 and 4 are assigned for crop cultivation. Because there is a considerably higher need for water in the process of crop cultivation (estimated at 3600l per day), initial setup requires higher capacities of storages and higher initial level of resources compared to the human habitation units:

- 1 storage unit each, 10000l capacity
- Initial resource volume: 5000l each
- No human inhabitants
- 2 distributed resource inputs: WRS and ISRU analogies

Economy factor for human water consumption is derived from the Table 4, where a minimum requirement estimated by NASA is set to 27.7 l/p/d, rounded at 28 l/p/d. This equals to 56% from the initial requirement value (50 l/p/d), so the economy factor being used equals to 0.56.

Similarly, economy factor for the crop cultivation is calculated from the minimum requirement of 10 l for cultivation area of $1m^2$, which equals to about 56% of the default requirements (18 l/m^2). Therefore, the economy factor used in the simulation for the crop cultivation resource consumption is also set to 0.56.

Resource input consist of an initial resource levels of 14000 l of water distributed throughout the settlement units 1-4, and the equivalents to the WRS (water recycling unit) and ISRU (water collecting unit). The simulation parameters were set with the assumption that these devices do not operate perfectly, or uniformly – otherwise, there would be no need for simulation, and the resource management could be determined by the simple set of mathematical formulas. Therefore, the WRS units are set to operate randomly between 75% and 85% success rate for each time unit (between measured and hypothetical/designed values).

There are two different groups of ISRU units assumed for the purpose of the simulation. One group is assigned to the storage units in human habitation modules. These units are set to produce resource

values randomly between 150 l and 380 l for each time unit. Other group is assigned to the crop cultivation modules, producing random resource values between 5000 l and 7000 l for each time unit. With such semi-random parameters setup, each instance delivers different scenario.

One of the possible scenarios is described hereafter, and presented with Figure 40. Because of the unplanned circumstances, Unit 3 was depleted of resources in the time unit 13, and was without water until the end of the observed period. We assume here that with the lack of irrigation, crops in the unit 3 eventually died. Unit 4 was depleted of resources at time unit 24, Unit 1 at time 42 and Unit 2 at time 46. Moreover, at time unit 46, three of the habitat units (1-3) were depleted of resources at the same time. Therefore, a settlement is considered not self-sustainable in the observed time period and with the current setup, and the self-sustainability ended in the time unit 13 (in about 195 days after the landing). The same setup will be used in the SSSHS framework, with the goal of prolonging the self-sustainability within the observed time period of 780 days.

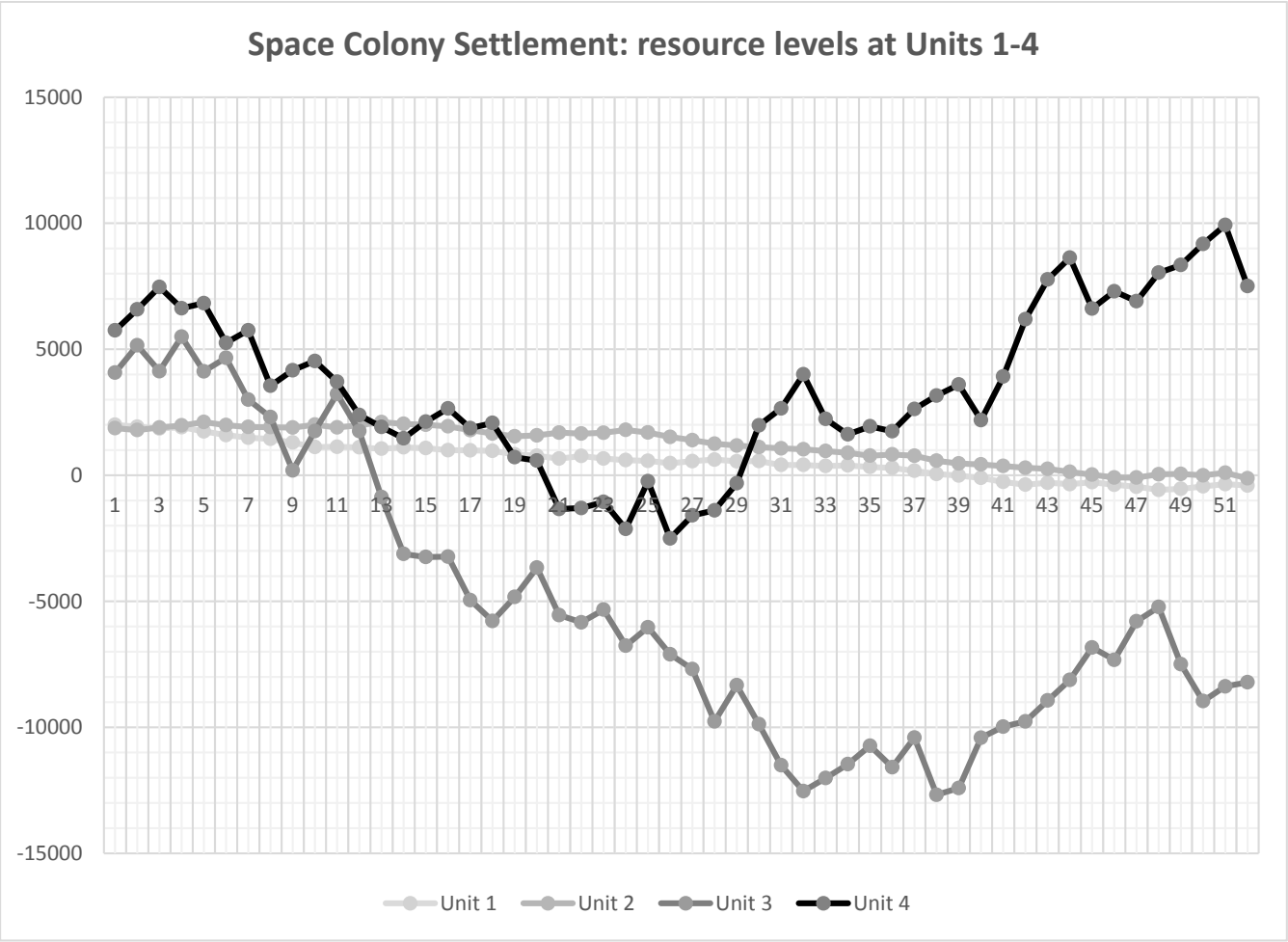

Figure 40. Space settlement habitat units 1-4; resource levels without the support of SSSHS infrastructure

6.3.1.2 Scenario 3, Simulation 1: Simulation Results and Discussion

The SSSHS simulation was run with the same initial setup as in the previously described static scenario, where the Mars settlement was not self-sustainable. The complete listing of the implementation of scenario 2 model is available in Appendix B. The results of the simulation are presented with the statistical simulation variables in framework Listing 10, and with the Figure 41.

```
 ******* Number of system interventions: 60
 ********** First intervention happened at time: 9


 ******* Number of LT ALERTS: 14

 ********** Number of DELAY  requests: 14
 ********** Number of ECONOMY requests: 13
 ********** Number of NEGOTIATION requests: 12


 ******* Number of UT ALERTS: 8

 ********** Number of RESTORE requests: 1
 ********** Number of ADVANCE requests: 0
 ********** Number of GIVE requests: 20
 ********** Overflow of resources: 8801.000000


INDIVIDUAL REPORT FOR STORAGE UNIT-1
 - CRL: 801
 - UT alerts: 2
 - Resources lost: 0.000000
 - LT alerts: 3
 - Economy reqs: 3
 - Delay reqs: 3
CRL HISTORY: [2000, 2002, 1925, 1854, 1907, 1742, 1594, 1490, 1445, 1298, 3999, 3999,
3999, 3999, 3999, 3969, 3885, 3880, 3380, 3243, 3189, 3074, 3124, 3025, 2036, 2006, 1569,
1065, 801, 1395.0, 801.0, 801.0]
OVERFLOW per time unit: []


INDIVIDUAL REPORT FOR STORAGE UNIT-2
 - CRL: 801
 - UT alerts: 2
 - Resources lost: 0.000000
 - LT alerts: 0
 - Economy reqs: 0
 - Delay reqs: 0
```

```
CRL HISTORY: [2000, 1871, 1794, 1894, 1977, 2112, 1989, 1917, 1909, 1891, 2375, 3999,
3999, 3999, 3929, 3896, 3833, 3681, 3547, 3437, 3473, 3583, 3543, 3579, 3692, 3588, 3414,
3285, 3143, 3070, 2408.0, 801.0]
OVERFLOW per time unit: []


INDIVIDUAL REPORT FOR STORAGE UNIT-3
 - CRL: -48223
 - UT alerts: 3
 - Resources lost: 8801.000000
 - LT alerts: 11
 - Economy reqs: 10
 - Delay reqs: 11
CRL HISTORY: [5000, 4069, 5152, 4127, 5496, 4121, 4660, 3003, 2306, 12070.999999999998,
10000, 9749, 7999, 5381, 3181, 3060, 3075, 1350, 1001, 1952, 3119, 1233, 1001, 1506,
1001, 1730, 1001, 1001, -753, 684, 1001.0, -25232.0]
OVERFLOW per time unit: [8801.0]


INDIVIDUAL REPORT FOR STORAGE UNIT-4
 - CRL: 1001
 - UT alerts: 1
 - Resources lost: 0.000000
 - LT alerts: 0
 - Economy reqs: 0
 - Delay reqs: 0
CRL HISTORY: [5000, 5747, 6583, 7469, 6624, 6827, 5253, 5750, 3550, 4165, 7999, 7175,
6199, 5785, 5337, 5999, 6524, 5741, 5945, 4587, 4451, 2532, 2570, 2812, 1743, 3634, 1363,
2278, 2478, 3551, 5850, 5850]
OVERFLOW per time unit: []
```

Listing 10. SSSHS simulation output for the 3rd scenario

There were a total of 60 interventions of the SSSHS mechanisms, the first one happening at time unit 9 (about 135 days after the landing) - Unit 3 reaching the lower threshold with its resource level. This is consistent with the static analysis, where Unit 3 was the first one to deplete its resource reserves. Because the first intervention happened relatively early in the simulation run, initial settlement setup would need to be considered for different parameterization.

There were more low threshold alerts (14) than upper threshold alerts (8), referencing to the problems of frequent resource deficit. The resource levels show nearly continuous resource decline, and the need for the more steady resource input is apparent even without the SSSHS simulation.

The first observation with the upper threshold alerts data might indicate the possible problems with the (too) small storage capacities. More specifically, all four of the settlement units triggered upper threshold alerts, meaning that at certain time units each one of them were faced with the overflow of resources caused by the non-optimal storage capacities. Unit 3 lost a total of 8801 l of water. It might be deducted that if the maximum storage capacity of Unit 3 were to be increased, this overflow could have been distributed and stored throughout the settlement units, in contrast to the irrecoverable loss that occurred during the simulation. The simple experiment showed that, by increasing the maximum capacity of the Unit 3 from the initial 10000 l to 20000 l, the total resource loss was diminished to zero liters, but the self-sustainability was additionally prolonged for only 1 time unit. Although all of the units eventually triggered upper threshold alerts, the total number of upper threshold alerts was relatively small, which also might indicate that increasing the storage capacities would not lead to the relevant changes in the system resource allocation dynamics and, consequently, to the self-sustainability time period.

The chosen granularity of the time units (15 days per time unit) could provide the first relevant insight into the possible outcomes of the mission, and deliver attention to the possible problems that would need to be addressed in order for the settlement to achieve self-sustainability. However, the more detailed analysis would require smaller granularity of the simulation, dealing with more precise time units and resource values throughout these units.

The results of the simulation showed that by using the SSSHS mechanisms, the self-sustainability of the settlement was prolonged until the time period 32 (about 480 days after the landing). This is consistent with the Hypothesis H1, and therefore the Hypothesis H1 is here proven true for the second scenario.

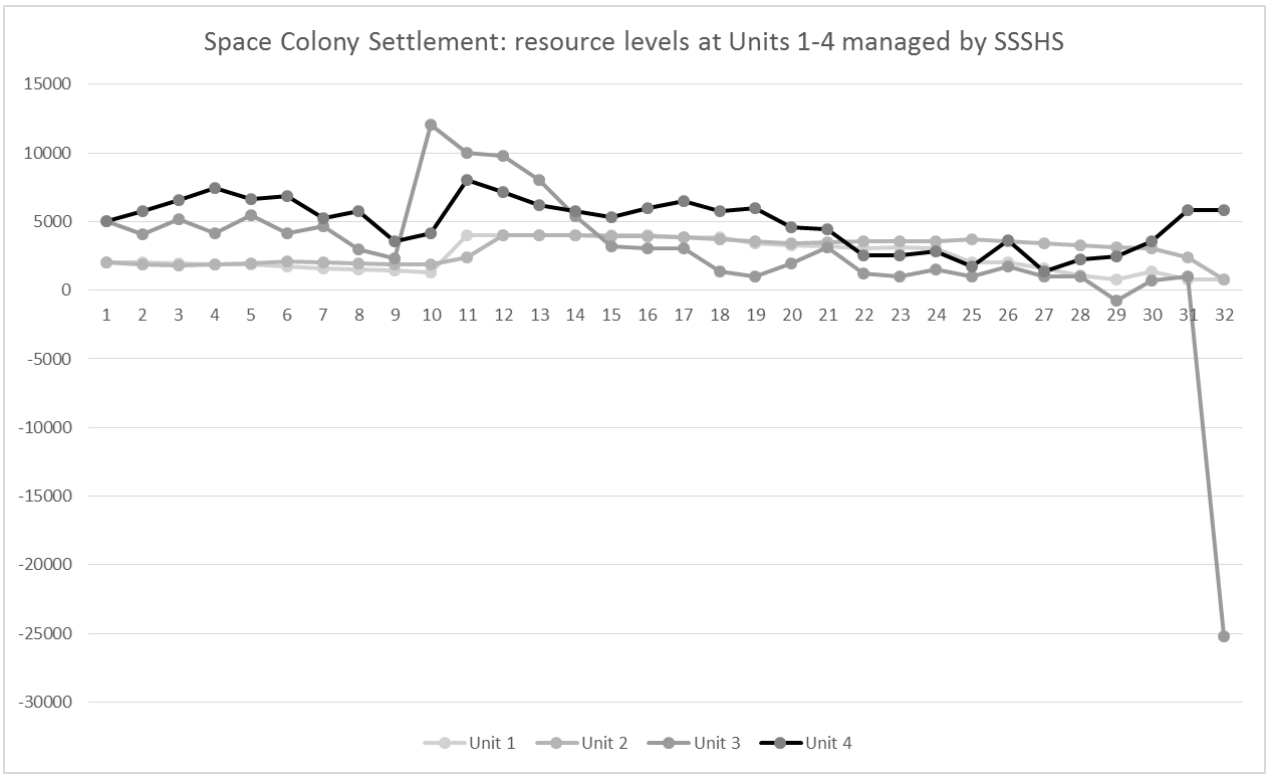

Figure 41. Space settlement habitat units 1-4; resource levels managed by SSSHS mechanisms

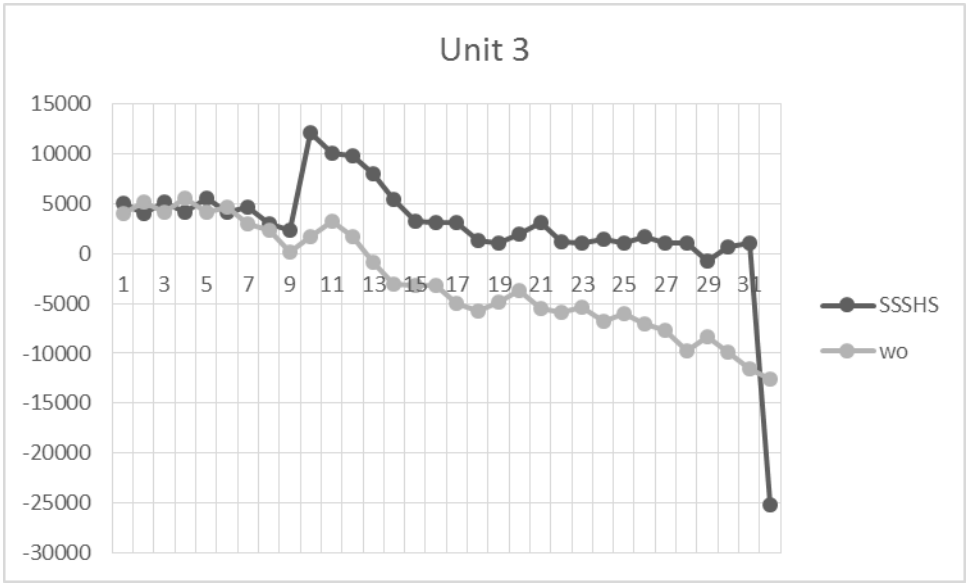

Figure 42. Resource levels with and without SSSHS mechanisms at Unit 3

Space colony missions which include human settlement units are different from Earth settlements in numerous parameters, and some of them which are relevant to the SSSHS framework are argued as follows:

- There is a fixed crew size for a planned time period, which could uniform the resource distribution consumption.
- There are fixed planned activities and schedules for the crew, resulting in a possibly insignificant variations in regarding resource consumption.
- Resupply from Earth is usually not feasible.
- Once the settlement is established, it is more difficult to execute subsequent structural changes.
- The extraction of the resource in-situ could be a subject of many unknown factors, resulting in the highly uncertain resource input distribution.

Considering these assumptions, agent-based SSSHS context which assumes a distributed network of independent, but interconnected settlement units, would facilitate a more reliable form of the Mars settlement, which would actively maintain the self-sustainability of the system, partially immune to the possible input variations and failing points within problematic settlement units.

6.3.2 Scenario 3, Simulation 2: Space Travel

Space travel scenario considers mobile habitats (e.g. spaceships) in contrast to previous scenarios where fixed habitats (planet-based dwellings) were considered. However, analogy to the fixed habitats is argued with the fact that mobile habitats require the similar basic principles regarding resource production, storage and consumption, with implementation specifics which are not relevant for the level of simulation abstract in the context of this research. The scenario will assume artificial gravity environment (1G) within the space vessels to nullify any possible variations in the resource consumption of the human crew regarding the differences in gravity forces.

In the autonomous, artificially intelligent spacecraft Deep Space 1 project [218], authors argue that "*any plans for space stations, interplanetary craft, extraterrestrial settlements will have intelligent building techniques at their heart*" [218].

Long duration space flights are not trivial. With currently existing technology, travelling to Mars takes about 180 days, which requires a considerable amount of food, water, energy, and other resources. Consequently, long-distance manned space travel requires large storage spaces and additional efforts to escape the Earth's gravity in launch.

Scenario will consider a developing technology for inducing human stasis, or torpor state (a state in which physiological and metabolic activity is considerably slowed down), which will utilize medical advances in therapeutic hypothermia and total parenteral nutrition (the intravenous feeding of a person by all essential nutrients needed for a human body to function: a mixture containing amino acids, lipids, dextrose, vitamins, electrolytes, and trace elements. The viability of such idea was initialized by NASA, which funded a study conducted by Atlanta-based aerospace engineering organization SpaceWorks Enterprises. [226]

Direct benefits of such state which are curious for the SSSHS framework include the reduction in mission resources (consumables) and reduction in storage volumes. Other benefits include reduced pressurized volume designated for living quarters, minimized psychological challenges for the crew, increased radiation shielding, reduced number of heavy-lift launches, etc.

SSSHS space travel scenario will utilize the hypothetical possibility of inducing torpor state in an unexpected event of the depletion of resources, where a SSSHS mechanism “delay” could intervene, placing the endangered crew into torpor, conserving the resources, and possibly saving the lives of the crew.

Scenario will include hypothetical fleet consisting of five independent and mutually networked mobile dwelling units (space crafts), which are capable of docking onto each other by request (need, or an event). Docking mechanisms would enable the transfer of resources which are impossible or not feasible of transporting wirelessly.

The duration of the simulation will be set to 180 days (flight to Mars), as 45 time units (four-day units).

Space crafts will be referred to with designated labels SS1, SS2, SS3, SS4, and SS5. There is a different configuration for the space crafts, which are designed by one of three possible layout classes. The setup is given as the following:

SS1 (Class C1)

- 6-person crew operating from time units 1-10
- 4-person crew operating from time units 11-35
- 6-person crew operating from time units 36-45
- 1 water storage unit with maximum capacity 1200l
- Initial resource volume: 1000l
- Two resource inputs: WRS and ISRU analogies, working constantly throughout the duration of the mission

SS2 and SS3 (Class C2)

- 4-person crew operating from time units 1-6
- 2-person crew operating from time units 7-39
- 4-person crew operating from time units 40-45
- 1 water storage unit with maximum capacity 1000l

- Initial resource volume: 400l
- Two resource inputs: WRS and ISRU analogies, working constantly throughout the duration of the mission

SS3 and SS4 (Class C3)

- 2-person crew operating from time units 1-6
- 2-person crew operating from time units 7-39
- 2-person crew operating from time units 40-45
- 1 water storage unit with maximum capacity 1000l
- Initial resource volume: 400l
- Two resource inputs: WRS and ISRU analogies, working constantly throughout the duration of the mission

6.3.2.1 Scenario 3, Simulation 2: Simulation Results and Discussion

The initial, ideal setup was parameterized with slight excess of resources in each time unit. The resource production was considered with an ideal operation output from all the devices that produce water. This setup is presented with Figure 43.

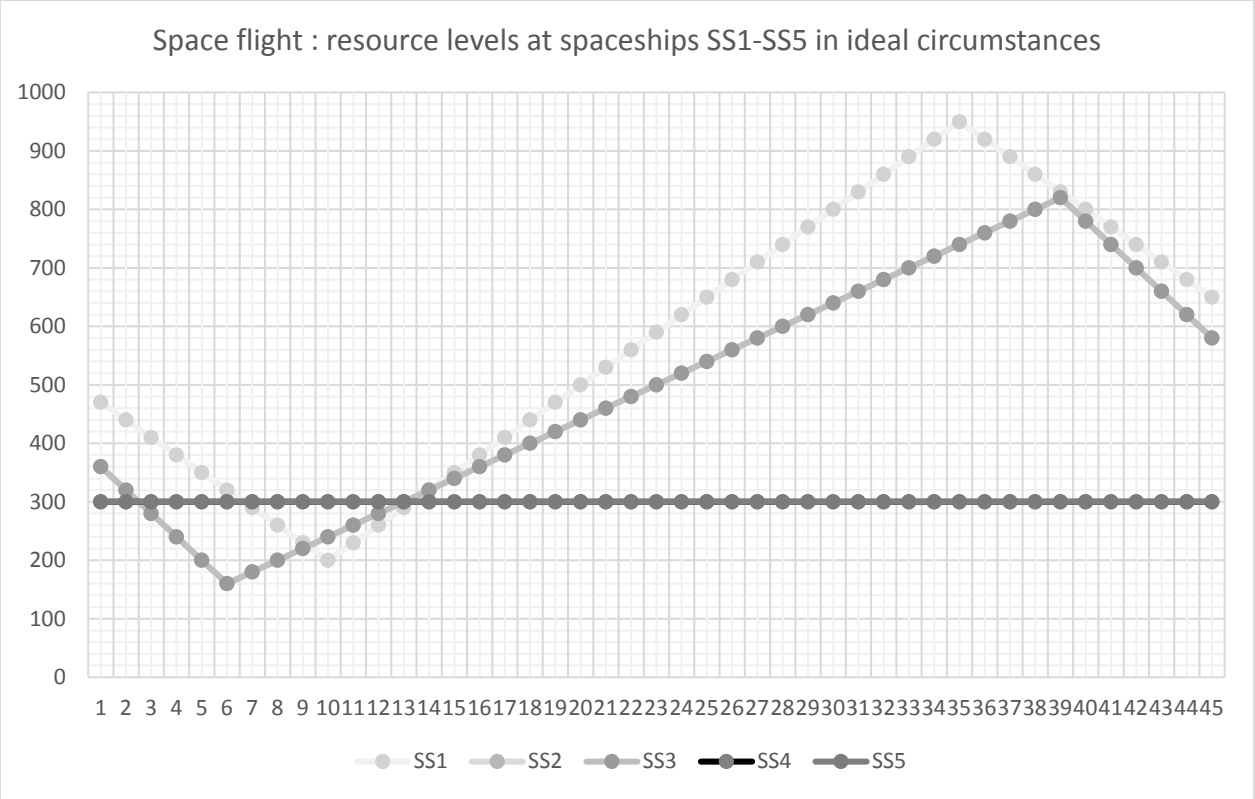

Figure 43. Resource levels in an ideal scenario

Adding uncertainties in resource production with implementing random numbers between defined ranges instead of using uniform distributions creates a different scenario, depicted with Figure 44.

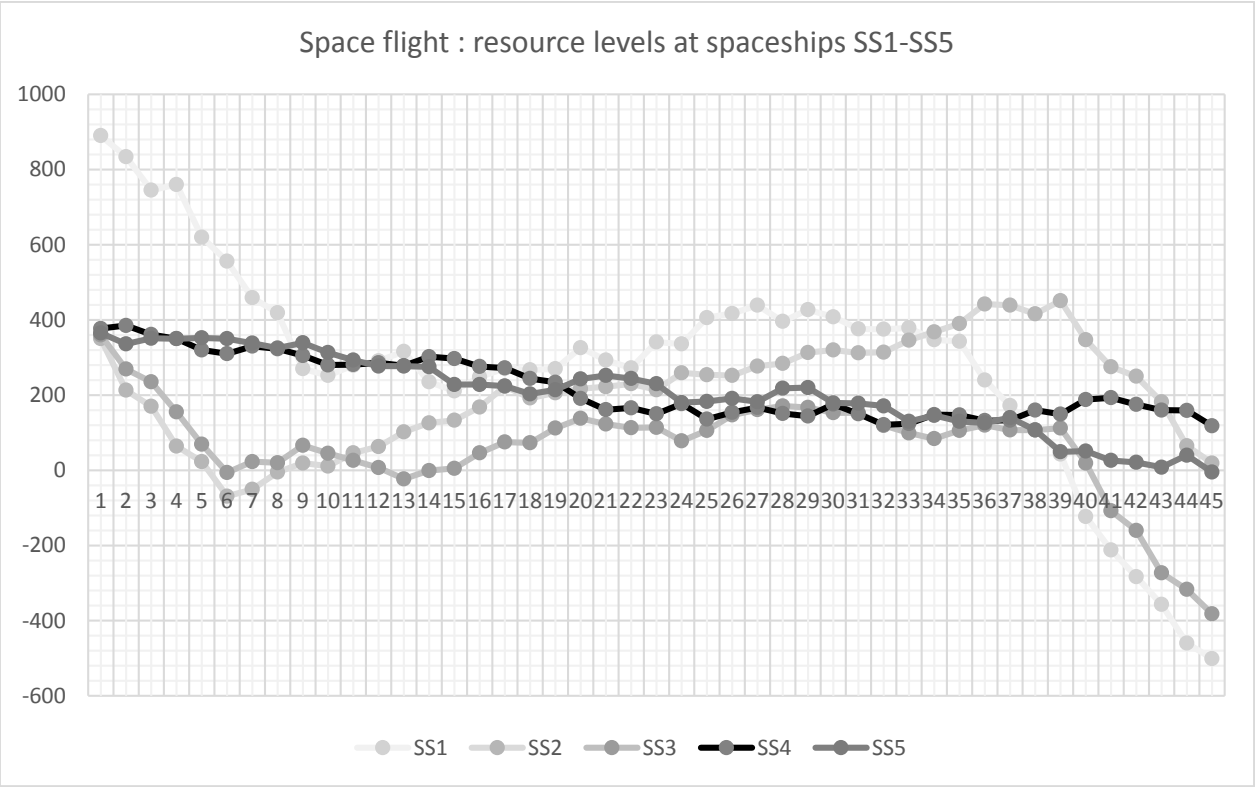

Figure 44. Resource levels in an unpredictable scenario

With these “unexpected” circumstances (a malfunction of the WRS device for example), the fleet would deplete its water resources, first at spaceships SS2 and SS3 (in time unit 6), then at SS1 (in time unit 40), and lastly, at SS5 (in time unit 45). In the context of this research, the fleet as a whole would become non-self-sustainable at time unit 6, or about 24 days after launch.

The SSSHS simulation was run with the same initial setup as in the previously described static scenario with random resource inputs, where the fleet ceased to be self-sustainable in time unit 6. The complete listing of the implementation of scenario 1 model is available in Appendix B. The results of the simulation are presented with the statistical simulation variables in the framework Listing 11, and with the Figure 45.

********** SYSTEM NOT SELF-SUSTAINABLE in time: 32 **********

.... [END OF SIMULATION] ....

******* Number of system interventions: 68
********** First intervention happened at time: 4

******* Number of LT ALERTS: 10

********** Number of DELAY requests: 10
********** Number of ECONOMY requests: 7
********** Number of NEGOTIATION requests: 6

******* Number of UT ALERTS: 23

********** Number of RESTORE requests: 0
********** Number of ADVANCE requests: 1
********** Number of GIVE requests: 44
********** Overflow of resources: 0.000000

INDIVIDUAL REPORT FOR STORAGE SS1
- CRL: -165
- UT alerts: 0
- Advance reqs: 0
- Resources lost: 0.000000
- LT alerts: 2
- Economy reqs: 2
- Delay reqs: 2

CRL HISTORY: [1000, 890, 834, 745, 760, 620, 556, 459, 419, 333, 315, 342, 354, 379, 322, 305, 342, 365, 361, 364, 420, 387, 366, 435, 430, 500, 511, 533, 490, 521, 502, 201.0]

INDIVIDUAL REPORT FOR STORAGE SS2
- CRL: 151

 - UT alerts: 15
 - Advance reqs: 1
 - Resources lost: 0.000000
 - LT alerts: 3
 - Economy reqs: 1
 - Delay reqs: 3
CRL HISTORY: [400, 350, 213, 170, 64, 23, 731, 749, 795, 799, 791, 799, 799, 799, 799, 799, 799, 799, 768, 782, 793, 798, 799, 783, 799, 794, 792, 799, 799, 799, 799, 232.99999999999997]

INDIVIDUAL REPORT FOR STORAGE SS3
 - CRL: -44
 - UT alerts: 8
 - Advance reqs: 0
 - Resources lost: 0.000000
 - LT alerts: 2
 - Economy reqs: 1
 - Delay reqs: 2
CRL HISTORY: [400, 360, 269, 235, 155, 69, 794, 799, 796, 799, 778, 759, 740, 710, 732, 738, 779, 799, 797, 799, 799, 784, 774, 775, 739, 767, 799, 799, 799, 795, 781, 381]

INDIVIDUAL REPORT FOR STORAGE SS4
 - CRL: -73
 - UT alerts: 0
 - Advance reqs: 0
 - Resources lost: 0.000000
 - LT alerts: 2
 - Economy reqs: 2
 - Delay reqs: 2
CRL HISTORY: [400, 377, 385, 361, 350, 320, 310, 354, 347, 329, 304, 332, 337, 329, 353, 348, 327, 332, 304, 331, 314, 284, 296, 280, 336, 295, 322, 353, 337, 359, 398, 173.99999999999997]

INDIVIDUAL REPORT FOR STORAGE SS5
 - CRL: 151
 - UT alerts: 0
 - Advance reqs: 0
 - Resources lost: 0.000000
 - LT alerts: 1
 - Economy reqs: 1
 - Delay reqs: 1

```
CRL HISTORY: [400, 365, 336, 351, 350, 352, 350, 338, 325, 339, 313, 293, 294, 333, 331,
284, 319, 370, 349, 360, 389, 398, 390, 376, 326, 329, 337, 342, 395, 397, 356, 151.0]
```

Listing 11. SSSHS simulation report for the 4th scenario

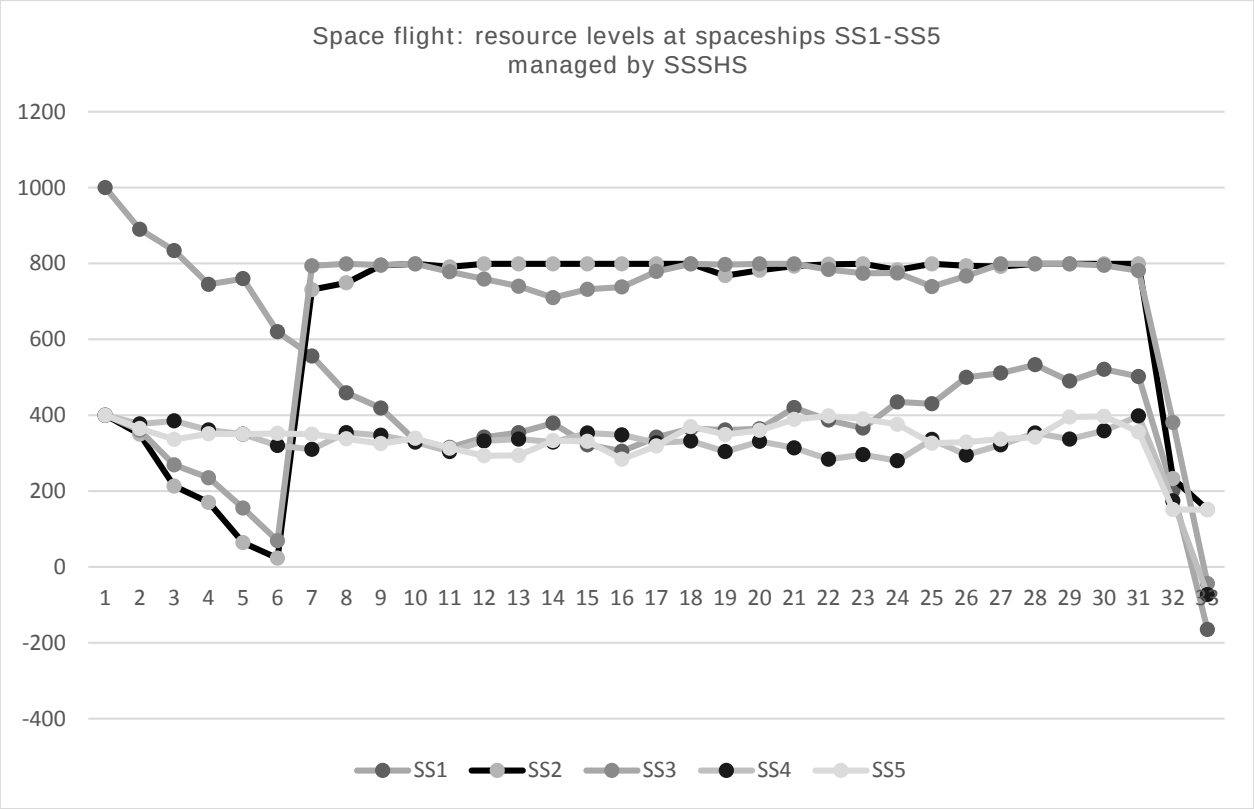

Figure 45. Resource levels managed by the SSSHS mechanisms

The following figure illustrates the difference within the system dynamics between using the SSSHS framework („SSSHS“) and without using the SSSHS framework („wo“). Within the „static“ analysis, the unit SS3 was depleted of resources in time unit 6, whereas the SSSHS framework mechanisms were able to prevent this event and maintain needed resource levels until the time unit 32.

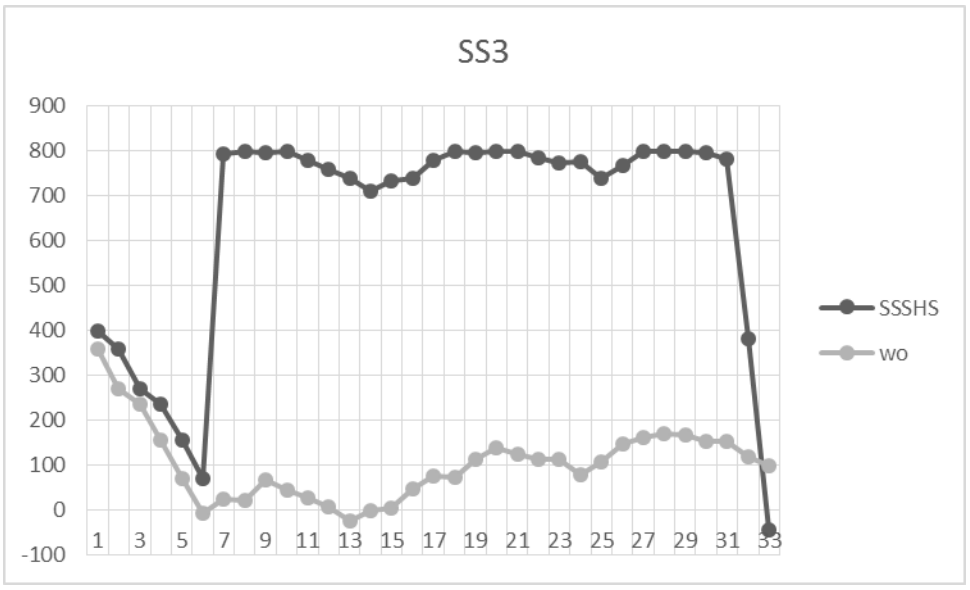

Figure 46. Resource levels with and without SSSHS mechanisms at SS3

The simulation terminated with a message “SYSTEM NOT SELF-SUSTAINABLE in time: 32”, which mean that the SSSHS framework tried to sustain the self-sustainability of the fleet as log as possible, but ultimately failed at time unit 32 (about 128 days after the launch) because of the general lack of resources – a total production of the resources in the simulated time period was considerably smaller that total consumption of the resources in the same time period.

There was a total of 68 SSSHS mechanisms’ interventions, with the first one (lower threshold alert at SS2) triggered at time unit 4, which is consistent with a static analysis of the system.
A total of 23 upper threshold alerts happened at SS2 (15) and SS3 (8), both of them belonging to the spaceship class C2. The number of “give” requests was 44, pointing to the large accumulation of resources at a certain time period. This could again point to the problem of inadequately dimensioned storage units, but an important variable reports that there were no resources lost at the end – they were relocated in time to ships that needed them.

Lower threshold alerts were triggered at least once at every spaceship at a given time, with a total of 10 of lower threshold alerts. Every mechanism was used to prolong the self-sustainability. Three of a total of five spaceships ended with negative values in their storages, and two of them were critically near their lower threshold values.

With the analysis of the simulation output, it can be concluded that there was an insufficient resource production during the mission duration, which could not cover even the minimum needs of the crew.

The results of the simulation showed that by using the SSSHS mechanisms, the self-sustainability of the space flight mission was prolonged until the time unit 32, where in the static analysis, the self-sustainability ended at time unit 6. This is consistent with the Hypothesis H1, and therefore the Hypothesis H1 is proven here as true.

7. Conclusion and Further Research

The SSSHS framework proved to prolong the self-sustainability of the settlements/fleet in all test-bed scenarios. The hypothesis H1 was therefore proven true, which is presented and argued in subsections 6.1.6, 6.2.3, 6.3.1.2, and 6.3.2.1 of this research.

The framework's reporting variables provide relevant insight into the nature of the developed models, focusing the attention on the possible problems with maintaining self-sustainability of the modeled system. The analysis based on the reporting variables should be performed in a holistic form, taking all the variables' values into considerations, their possible connections and the nature of such connections, to gain an objective aspect on the nature of the modeled system.

The results presented in this research are possible to recreate using different scenario configurations and running the simulations within the SSSHS framework, or creating a different set of test-bed scenarios. Introducing the uncertainty, or some form of semi-controlled randomness in the simulation input from the environment and/or output towards resource consumers, is a relevant factor for the SSSHS context; the framework is able to determine the course of action for each event or circumstances in every time unit, and trigger appropriate set of activities in order to reach its goal.

The relevant advantage of the proposed SSSHS resource management context is the resource sharing. While a single dwelling unit could implement a number of self-sustainability mechanisms, networking the resource consumers, and managing their consumption loads, the system could enter problematic scenario in preserving the self-sustainability because of the single point of failure. The networked system of more than one sub-units, which are independent and proactive, would considerably mitigate the effects of possible glitches in the system.

In the introductory chapter 1, self-sustainability in the research context was defined, and research goals and hypothesis was presented. Chapter 2 presented a more verbose research plan, starting from the conceptual framework towards building the framework with implementation and research methods' specifics.

In the chapter 3 an overview of the literature related to the context of this research was presented; research areas such as agents and multi-agent systems, intelligent buildings, smart cities, sustainable development, eco feedback, load management, and other related fields were introduced. Along with the literature overview, an unstructured interviews conducted with three experts on self-sustainability (introduced in section 4.1) delivered a significant input used in the conceptualization and the development of the SSSHS mechanisms. Those mechanisms form a core of the SSSHS framework, and are explained in detail in chapter 4, along with other SSSHS framework implementation specifics (implemented classes of agents, behaviours of agents, communication, methods, activity flow, etc.).

Test bed scenarios were developed for the purpose of testing the research hypothesis, and are presented in chapter 6. The following scenarios were considered and simulated:

1. A permaculture, water-management scenario, developed from the gathered data on the location of interest. A case study suggested that the observed eco-village were depleted of water before the end of the observed time period. An SSSHS framework simulation inherited the same input data, and managed to sustain self-sustainability until the end of the observed time period.

2. An energy production process using photovoltaic solar modules and auxiliary propane generators presented a complex ground for developing a scenario with the possibility to simultaneously simulate more than one resource, and consequently, more than one storage types per dwelling unit (energy stored as watt-hours, and propane stored in liters). Because of the specific, existing technologies involved in this scenario, an expert in the domain of off-grid power producing was consulted, and the scenario was developed under his supervision. The results of the simulated scenario within the SSSHS framework environment showed that the framework was able to prolong the self-sustainability of the settlement in two different simulations – one carried out with using propane as an auxiliary power source, equivalently to the scenario observed in a static analysis, and the other, with the same initial setup, but without using the propane. The framework also showed an ability

to manipulate the battery usage patterns, which would require further analysis and experimentations.

3. A human habitat on Mars scenario was partially developed from the documents available on the Mars One mission feasibility analysis. A static scenario was developed in spreadsheets, using the data available from the mission analysis, and by adding a randomness in values which presented the resource production, by assuming that the devices used for resource production would not operate at their 100% efficiency, or that the environment would not provide the exact amount of resources predicted in the mission plan. Although the static analysis of one of the possible scenario configurations showed that the Mars human habitat would become depleted of resources at time unit 13, the SSSHS framework was able to maintain the self-sustainability until the time unit 32.

4. The most hypothetical scenario involved a prolonged space travel of the human crew. Research into human stasis initiated by NASA was referenced as a basis for the operational delaying self-sustainability mechanism, and other parameters were derived with the assumption of artificial gravity and other Earth-like environment conditions which would facilitate similar resource consumption dynamics as on Earth-based human settlements. The physical resource production mechanisms were argued with the water-recycling units and hypothetical water production units. Values for both of these production units were randomized within the defined end-values. The static analysis of the fleet was performed on such randomly generated values within spreadsheets, and showed cessation of the self-sustainability of the fleet at time unit 6. The simulation was run in the SSSHS framework with the same setup, and the framework was able to prolong the self-sustainability of the fleet until the time unit 32. Although the self-sustainability of the fleet was unable to hold until the end of the observed time period in this specific model setup, the framework was able to significantly prolong the self-sustainability, and reported on the results of the simulation.

Limitations related to scenarios three and four could be observed in the lack of domain-specific and more accurate technical details of the technology implementation mechanisms, physical space-

related parameters and similar factors which might interfere with the self-sustainability within these scenarios. However, because the SSSHS framework works with final number values of such factors and mechanisms to which the specific meanings are attached in the scenario implementation process, such factors can be straightforwardly assigned to the framework scenario parameters by the experts on the fields of space travel, colonies, and similar fields of research.

The future SSSHS framework research will consider including additional self-sustainability mechanisms, such as “skip” function, for instance. This function would enable the sub-system to completely skip a certain resource consumer which would be previously identified as non-critical to the operation of the system. This would aid in critical events of depletions of resources where all the other mechanisms were already previously activated, but with no desirable outcome.

A few more methods would be considered for a further research and development. “Forecasting” method would continuously read input data from the past resource dynamics, and would try to determine the resource level trends and trigger the self-sustainability mechanisms accordingly. This method could also read real forecasting data from another system (e.g. meteorological forecast) in order to optimize the actions in its sub-systems.

Agent-based modeling is a natural methodology to analyze and build such systems, given the large number of non-deterministically interacting components (agents) including but not limited to humans, various sensor and actuator equipment, appliances, as well as environmental influences.

The developed SSSHS framework was intentionally and significantly simplified by the use of benevolence assumption on cooperative problem solving. Future research may include an assumption about non-cooperative agents, where a more complex nature of agent cooperation, negotiation and conflict resolution would need to be considered for implementation. Scenarios with domestic resource usage management could potentially benefit in terms of efficiency of resource allocation by implementing learning strategies in agents.

Learning might prove to be useful in adaptation of the agents to new, previously unexperienced situations, where an agents could update their own decision rules based on the past experiences.

For example, this might include autonomous updating of resource threshold values, resulting in an adapted decision on when to activate the self-sustainability mechanisms. Also, the agents might recognize important patterns in key repeating events (e.g. summer droughts) based on the past experiences, and adjust their behaviours and decisions in the future accordingly.

By taking these notices into consideration, the potential for further extending the research based on the SSSHS framework is clearly broad and significant.

References

1. Moriarty, P., & Honnery, D. Intermittent renewable energy: The only future source of hydrogen?. *International Journal of Hydrogen Energy*, *32*(12), 1616-1624., 2007

2. Ni, M., Leung, D. Y., Leung, M. K. A review on reforming bio-ethanol for hydrogen production. *International Journal of Hydrogen Energy*, *32*(15), 3238-3247., 2007

3. Bond, A. H. and Gasser, L. (eds). Readings in Distributed Artificial Intelligence. Morgan Kaufmann, San Mateo, CA., 1988

4. Jennings, N. R. Wooldridge, M. Applications of intelligent agents. In Agent Technology: Foundations, Applications and Markets (eds. N. R. Jennings and M. Wooldridge), pp. 3-28. Springer, Berlin., 1998

5. Wooldridge, M. An Introduction to MultiAgent Systems. Wiley, 2009

6. Wooldridge, M. Intelligent agents. In: Weiss, G. (Ed.), Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence. MIT Press, pp. 27–78., 1999

7. Cockburn, A. Structuring Use Cases with Goals. Journal of Object-Oriented Programming, Sep-Oct, 1997 and Nov-Dec, 1997

8. DeLoach, S. A., Wood, M. F., & Sparkman, C. H. Multiagent systems engineering. International Journal of Software Engineering and Knowledge Engineering, 11(03), 231-258., 2001

9. Miller, G., Spoolman, S. Living in the environment: principles, connections, and solutions. Cengage Learning, 2011

10. Gleick, P. H. Basic water requirements for human activities: Meeting basic needs. Water international, 21(2), 83-92., 1996

11. Wikipedia contributors. Water resources. Wikipedia, The Free Encyclopedia. Retrieved May 18, 2015, from: https://en.wikipedia.org/w/index.php?title=Water_resources&oldid=667354474

12. Electricity [Def. 1, 2]. (n. d.). In Oxford Dictionaries. Retrieved January 12, 2014, from: http://www.oxforddictionaries.com/definition/english/electricity

13. Wikipedia contributors. Renewable energy. Wikipedia, The Free Encyclopedia. Retrieved May 22, 2015, from: https://en.wikipedia.org/w/index.php?title=Renewable_energy&oldid=666737211

14. Zalba, B., Marín, J. M., Cabeza, L. F., Mehling, H. Review on thermal energy storage with phase change: materials, heat transfer analysis and applications. Applied thermal engineering, 23(3), 251-283., 2003

15. Wikipedia contributors. Food. Wikipedia, The Free Encyclopedia. Retrieved June 18, 2015, from https://en.wikipedia.org/w/index.php?title=Food&oldid=666410978.

16. Deelstra, T., Girardet, H. Urban agriculture and sustainable cities. Bakker N., Dubbeling M., Gündel S., Sabel-Koshella U., de Zeeuw H. Growing cities, growing food. Urban agriculture on the policy agenda. Feldafing, Germany: Zentralstelle für Ernährung und Landwirtschaft (ZEL), 43-66., 2000

17. Alliance, S. A. F. E. The Food Miles Report: The dangers of long distance food transport. London: SAFE Alliance., 1994

18. Jarosz, L. The city in the country: Growing alternative food networks in Metropolitan areas. Journal of Rural Studies, 24(3), 231-244., 2008

19. National Aeronautics and Space Administration, International Space Station Environmental Control and Life Support System. Retrieved March 12, 2015, from: http://www.nasa.gov/centers/marshall/pdf/104840main_eclss.pdf

20. Marik, V., Stepankova, O., Krautwurmova, H., Luck, M. (Eds.). Multi-Agent-Systems and Applications II: 9th ECCAI-ACAI/EASSS 2001, AEMAS 2001, HoloMAS 2001. Selected Revised Papers (Vol. 2322). Springer, 2003

21. Dell'Acqua, P., Engberg, M., Pereira, L. M. An architecture for a rational reactive agent. In Progress in Artificial Intelligence (pp. 379-393). Springer Berlin Heidelberg, 2003

22. Silva, A., Da Silva, M. M., Delgado, J. An overview of AgentSpace: a next-generation mobile agent system. In Mobile Agents (pp. 148-159). Springer Berlin Heidelberg, 1998

23. d'Inverno, M., Luck, M., Luck, M. M. Understanding agent systems. Springer Science & Business Media, 2004

24. Martins, R. M., Ribeiro Chaves, M., Pirmez, L., Fernando Rust da Costa Carmo, L. Mobile agents applications. Internet Research, 11(1), 49-54, 2001

25. Wagner, G. Towards agent-oriented information systems. Preliminary Report, 1999

26. Hindriks, K. V., De Boer, F. S., van der Hoek, W., Meyer, J. J. C. A programming logic for part of the agent language 3APL. In Formal Approaches to Agent-Based Systems (pp. 78-89). Springer Berlin Heidelberg, 2001

27. Castillo, A.B. Improving robustness in robotic navigation by using a self-reconfigurable control system. PhD diss., Master's thesis, Facultad De Informática Universidad Politécnica

De Madrid, 2011

28. Kowalski, R. A. Using meta-logic to reconcile reactive with rational agents. Meta-logics and logic programming, 227-242, 1995

29. Genesereth, M. R., Nilsson, N. J. Logical Foundations of Artificial Intelligence. Morgan Kaufmann, 1987

30. Tropism [Def. 1]. (n. d.). In Oxford Dictionaries. Retrieved January 13, 2014, from: http://www.oxforddictionaries.com/definition/english/tropism

31. Feijó, B., Gomes, P. C. R., Bento, J., Scheer, S., Cerqueira, R. Distributed agents supporting event-driven design processes. In Artificial Intelligence in Design'98 (pp. 557-577). Springer Netherlands, 1998

32. Lin, Z., Carley, K. Proactive or reactive: An analysis of the effect of agent style on organizational decision making performance., 1993

33. Larson, L. L., Bussom, R. S., Vicars, W., Jauch, L. Proactive versus reactive manager: is the dichotomy realistic? Journal of Management Studies, 23(4), 385-400., 1986

34. Wooldridge, M. Conceptualising and Developing Agents. Proceedings of the UNICOM Seminar on Agent Software. London, 1995

35. Hayzelden, A. L.; Bigham J. Software agents for future communication systems. 1st ed. New York: Springer, Pp. 101., 1999

36. Jung, C. G., & Fischer, K. Logic-Based Hybrid Agents. In Computational Logic: Logic Programming and Beyond (pp. 626-654). Springer Berlin Heidelberg, 2002

37. Lieberman, H. Interfaces that give and take advice. Human-Computer Interaction for the New Millenium, 475-485., 2001

38. Lieberman, H., Selker, T. Agents for the user interface. Handbook of Agent Technology, 1-21., 2003

39. Serenko, A., Bontis, N., Detlor, B. End-user adoption of animated interface agents in everyday work applications. Behaviour & Information Technology, 26(2), 119-132., 2007

40. Driankov, D., Saffiotti, A. Fuzzy Logic Techniques for Autonomous Vehicle Navigation. Springer-Verlag, 2001

41. Wooldridge, M., Jennings, N. R. Formalizing the cooperative problem solving process. Synthese Library, 143-162., 1997

42. Smith, R. G., Davis, R. Frameworks for cooperation in distributed problem solving. Systems, Man and Cybernetics, IEEE Transactions on, 11(1), 61-70, 1981

43. Worrest, W. Conflict-resolution strategies for nonhierarchical distributed agents. Distributed artificial intelligence, 2, 139., 1989

44. Galliers, J. R. A theoretical framework for computer models of cooperative dialogue, acknowledging multi-agent conflict. Doctoral dissertation, Open University, 1988

45. Lander, S., Lesser, V. R., Connell, M. E. Conflict resolution strategies for cooperating expert agents. In CKBS'90 (pp. 183-200). Springer London, 1991

46. Ephrati, E., Rosenschein, J. S. Multi-agent planning as a dynamic search for social consensus. In IJCAI (Vol. 93, pp. 423-429), 1993

47. Rosenschein, J. S., Zlotkin, G. Rules of encounter: designing conventions for automated negotiation among computers. MIT press, 1994

48. Berndtsson, M., Chakravarthy, S., Lings, B. Coordination among agents using reactive rules. Technical Report HS-IDA-TR-96-011, Department of Computer Science, University of Skövde, 1996

49. Smith, R. G. The contract net: A formalism for the control of distributed problem solving. In Proceedings of the 5th international joint conference on Artificial intelligence-Volume 1 (pp. 472-472). Morgan Kaufmann Publishers Inc., 1977

50. Smith, R. G. The contract net protocol: High-level communication and control in a distributed problem solver. IEEE Transactions on computers, (12), 1104-1113., 1980

51. Faratin, P., Sierra, C., Jennings, N. R. Negotiation decision functions for autonomous agents. Robotics and Autonomous Systems, 24(3), 159-182., 1998

52. Pruitt, D. G. Negotiation behavior. Academic Press, 1981

53. Zgrzywa, M. Consensus Determining with Dependencies of Attributes with Interval Values. J. UCS, 13(2), 329-344, 2007

54. Mencke, S., Dumke, R. Agent Supported E-learning. Univ., Fak. für Informatik., 2007

55. Aïmeur, E. Strategic Use of Conflicts in Tutoring Systems. In Conflicting Agents: Conflict Management in Multi-Agent Systems (pp. 223–250). Kluwer Academic Publishers, Norwell, MA, USA., 2001

56. Huhn, M., Duraslan, A., Gănceanu, G., Görmer, J., Hähner, J., Müller, J. P., ... & Schulz, C. Autonomous agents in organized localities: Metamodel and conceptual architecture. NTH Research School for IT Ecosystems, Clausthal University of Technology, Tech. Rep, 2,

2010.

57. Jung, H., Tambe, M. Conflicts in agent teams. In *Conflicting agents* (pp. 153-167). Springer US, 2002

58. Chantemargue, F. Conflicts in Collective Robotics. In *Conflicting Agents* (pp. 203-220). Springer US, 2002

59. Tessier, C., Laurent, M., Fiorina, H. , Chaudron, L. Agents' conflicts: new issues. In *Conflicting Agents* (pp. 1–32), 2002

60. Hannebauer, M. Their problems are my problems. In *Conflicting Agents* (pp. 63–100), 2002

61. Malsch, T., Weiss, G. Conflicts in social theory and MAS. In *Conflicting Agents* (pp. 101–152), 2002

62. Tomlin, C., Pappas, G. J., Sastry, S. Conflict resolution for air traffic management: A study in multiagent hybrid systems. *Automatic Control, IEEE Transactions on*, *43*(4), 509-521, 1998

63. Resmerita, S., Heymann, M. Conflict resolution in multi-agent systems. In *IEEE Conference on Decision and Control* (Vol. 3, pp. 2537-2542)., 2003

64. Lian, J., Shatz, S. M. A modeling methodology for conflict control in multi-agent systems. *International Journal of Software Engineering and Knowledge Engineering*, *18*(03), 263-303., 2008

65. Liu, T. H., Goel, A., Martin, C. E., Barber, K. S. Classification and representation of conflict in multi-agent systems., 1998.

66. Muñoz Ortega, A., Botía Blaya, J. A. A formal model of persuasion dialogs for interactions among argumentative software agents. Journal of Physical Agents, 3(3), 3-10, 2009

67. Revesz, P. Z. On the semantics of theory change: arbitration between old and new information. In *Proceedings of the twelfth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems* (pp. 71-82). ACM, 1993.

68. Cholvy, L. Reasoning about merging information. In D. Gabbay and P. Smets, editors, Handbook of Def. Reas. and Unc. Management Systems, volume 3, pages 233–263. Springer, 1998

69. Konieczny, S., Perez, R.P.. On the logic of merging. In Proc. KR, pages 488–498. Morgan Kaufmann, 1998

70. Benferhat, S., Dubois, D., Kaci, S., Prade, H. Possibilistic merging and distance-based fusion of propositional information. Annals of Mathematics and AI, 34(1–3):217–252, 2002

71. Chesñevar, C., Maguitman, A.G., Simari, G.R. Argument-based critics and recommenders: a qualitative perspective on user support systems. Data Knowledge Eng., 59(2):293–319, 2006

72. Besnard, P., Hunter A. Elements of Argumentation. MIT Press, 2008

73. Rahwan, I., Simari, G.R., editors. Argumentation in Artificial Intelligence. Springer, 2009

74. Bench-Capon, T., Prakken, H., Sartor, G. Argumentation in legal reasoning. In Rahwan and Simari [73], pages 363–382, 2009

75. Nguyen, N.T. Advanced Methods for Inconsistent Knowledge Management (Advanced Information and Knowledge Processing). Secaucus, NJ, USA: Springer-Verlag New York,

Inc., 2007

76. Fox, J., Krause, P., Ambler, S. Arguments, contradictions and practical reasoning. In *Proceedings of the 10th European conference on Artificial intelligence* (pp. 623-627). John Wiley & Sons, Inc., 1992

77. Huhns, M.N., Bridgeland, D.M. Multiagent truth maintenance. IEEE Transactions on Systems, Man and Cybernetics, vol. 21, no. 6, pp. 1437–1445, 1991

78. Kraetzschmar, G.K. Distributed Reason Maintenance for Multiagent Systems, J. Siekmann and J. G. Carbonell, Eds. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1997

79. Ram, S. Park, J. Semantic conflict resolution ontology (SCROL): An ontology for detecting and resolving data and schema-level semantic conflicts. Transactions on Knowledge and Data Engineering, vol. 16, no. 2, pp. 189–202, 2004

80. Lee, K. Kim, M. A novel approach for conflict resolution in context-awareness using semantic unification of multi-cognition. In PRIMA ’08: Proceedings of the 11th Pacific Rim International Conference on MultiAgents. Berlin, Heidelberg: Springer-Verlag, pp 267-274, 2008

81. Milind Tambe. Beliefs, Desires, Intentions (BDI). Class Resource. Retrieved May 17, 2015 from: https://www.cs.drexel.edu/~greenie/cs510/bdilogic.pdf

82. Rumbaugh, J., Jacobson, I. Booch, G. The Unified Modelling Language Reference Manual, Second edition, Addison,-Wesley., 2004

83. Schreiber, G., Akkermans, H., Anjewierden, A. Hoog, R. de, Shadbolt, N., Velde, W.Van de, Wielinga, B. Knowledge Engineering and Management: The CommonKADS Methodology, Cambridge, MA, ISBN: 0262193009, 1999

84. Odell, J., Parunak, H., Bauer, B. Representing agent interaction protocols in UML, In: First International Workshop on Agent-Oriented Software Engineering, (AOSE 2000), Ciancarini, P., Wooldridge, M., Eds., LNCS 1957 Springer, Limerick, Ireland, 121-140., 2001

85. Iglesias, C.A. González, J.C. A Survey of Agent-Oriented Methodologies In Proceedings of the 5th International Workshop on Agent Theories, Architectures and Languages (ATAL'98),LNAI n1555, Springer Verlag, Paris, France, 317-330., 1998

86. Henderson-Sellers, Brian Giorgini, Paolo (ed). Agent-oriented Methodologies. 1ed: Idea Group Inc, London, UK, ISBN 1-59140-581-5, p412., 2005

87. Bauer, B., Müller, J. P., Odell, J. Agent UML: A formalism for specifying multiagent software systems. International journal of software engineering and knowledge engineering, 11(03), 207-230., 2001

88. Burmeister, B. Models and methodology for agent-oriented analysis and design. Working Notes of the KI, 96(96-06), 52., 1996

89. Kendall, E. A. Agent software engineering with role modelling. In Agent-Oriented Software Engineering (pp. 163-169). Springer Berlin Heidelberg, 2001

90. Kinny, D., Georgeff, M., Rao, A. A methodology and modelling technique for systems of BDI agents. In Agents breaking away (pp. 56-71). Springer Berlin Heidelberg, 1996

91. Omicini, A. SODA: Societies and infrastructures in the analysis and design of agent-based systems. In Agent-oriented software engineering (pp. 185-193). Springer Berlin Heidelberg., 2001

92. Brazier, F. M., Keplicz, B. D., Jennings, N. R., Treur, J. Formal specification of multi-agent systems., 1995

93. Collinot, A., Drogoul, A., & Benhamou, P. Agent oriented design of a soccer robot team. In Proceedings of the Second International Conference on Multi-Agent Systems (ICMAS-96) (pp. 41-47)., 1996

94. Luck, M., Griffiths, N., & d'Inverno, M. (1997). From agent theory to agent construction: A case study. In Intelligent Agents III Agent Theories, Architectures, and Languages (pp. 49-63). Springer Berlin Heidelberg.

95. Sudeikat, J., Braubach, L., Pokahr, A., Lamersdorf, W. Evaluation of agent–oriented software methodologies–examination of the gap between modeling and platform. In Agent-Oriented Software Engineering V (pp. 126-141). Springer Berlin Heidelberg, 2005

96. Argente, E., Julian, V., Botti, V. Multi-agent system development based on organizations. Electronic Notes in Theoretical Computer Science, 150(3), 55-71., 2006

97. Wood, M. F., DeLoach, S. A. An overview of the multiagent systems engineering methodology. In Agent-Oriented Software Engineering (pp. 207-221). Springer Berlin Heidelberg., 2001

98. Shokooh, S., Khandelwal, T., Shokooh, F., Tastet, J., Dai, J. J. Intelligent load shedding need for a fast and optimal Solution. IEEE PCIC Europe, 1-6., 2005

99. United Nations General Assembly. Report of the world commission on environment and development: Our common future, chapter 2: Towards sustainable development. Technical report, 1987

100. Elkington, J. Enter the Triple Bottom Line. In Henriques, A. and Richardson, J. (eds.) The Triple Bottom Line, Does it All Add Up?: Assessing the Sustainability of Business and CSR. London: Earthscan, pp.1-16, 2004

101. Harris, F. Sustainable Development. In Harris, F. (ed.) Global Environmental Issues. West Sussex: John Wiley and Sons Ltd., 2004

102. Scott, A. (ed.). Dimensions of Sustainability: architecture, form, technology, environment, culture. London: E & FN Spon, 1998

103. Johnston, P., Everard, M., Santillo, D, and Robèrt, K.-H. Reclaiming the definition of sustainability. Environmental science and pollution research international, 14 (1):60–66, 2007

104. Gladwin, T., Kennelly, J., and Krause, T. Shifting Paradigms for Sustainable Development: Implications for Management Theory and Research. Academy of Management Review (6:2), pp. 874-907., 1995

105. Sustainable [Def. 1, 2, 3]. (n. d.). In Merriam Webster Online. Retrieved January 10, 2014, from http://www.merriam-webster.com/dictionary/sustainable?show=0&t=1390133123

106. Sustainable [Def. 1, 2]. (n. d.). In Cambridge Dictionaries Online. Retrieved January 10, 2014, from http://dictionary.cambridge.org/dictionary/british/sustainable?q=sustainable

107. Self-sustaining [Def. 1]. (n.d.). In Merriam Webster Online. Retrieved January 10, 2014, from http://www.merriam-webster.com/dictionary/self-sustaining

108. Self-sustaining [Def. 1]. (n.d.). In TheFreeDictionary.com. Retrieved January 10, 2014, from http://www.thefreedictionary.com/self-sustaining

109. Robèrt, K.-H. Educating a nation: The natural step. IN CONTEXT, 28:2–12, 1991

110. Hoffert, M., Caldeira, K., Benford, G., Criswell, D., Green, C., Herzog, H., Jain, A., Kheshgi, H., Lackner, K., and Lewis, J. Advanced Technology Paths to Global Climate Stability: Energy for a Greenhouse Planet. Science (298:5595), p p. 981-987., 2002

111. Hopkins, R. J. Localisation and resilience at the local level: the case study of Transition Town Totnes (Devon, UK) (Doctoral dissertation), 2010

112. Anthony, J. Family self-sufficiency programs - An evaluation of program benefits and factors affecting participants' success. Urban Aff. Rev. 2005, 41, 65-92., 2005

113. Lindbergh, L., Larsson, C.G., Wilson, T.L. Cost control and revenue generation: The case of public-housing companies' experiences in Sweden. Reg. Stud. 2004, 38, 803-815., 2004

114. Wackernagel M., Rees W. Our ecological footprints. Gabriola Island, BC, Canada: New Society Publishers, 1996

115. Nguyen H., Yamamoto R. Modification of ecological footprint evaluation method to include non-renewable resources consumption using thermodynamic approach. Resources, Conservation and Recycling 51.4 (2007): 870-884., 2007

116. Ellegard K., Palm J. Visualizing energy consumption activities as a tool for making everyday life more sustainable. Applied Energy, 88(5), 1920-1926., 2011

117. Ros J., Nagelhout D., Montfoort J. New environmental policy for system innovation: casus alternatives for fossil motor fuels. Applied Energy 86.2 (2009): 243-250., 2009

118. Diakaki C., Grigoroudis E., Kabelis N., et al. A multi-objective decision model for the improvement of energy efficiency in buildings. Energy 2010;35:5483–96., 2010

119. Kabashi S., Bekteshi S., Ahmetaj S., et al. Effects of Kosovo's energy use scenarios and associated gas emissions on its climate change and sustainable development. Applied Energy, 88(2), 473-478., 2011

120. Carvalho M.G., Bonifacion M., Dechamps P. Building a low carbon society. Energy 2011;36:1842–7., 2011

121. Lam H.L., Varbanov P.S., Klemes J.J. Minimising carbon footprint of regional biomass supply chains. Resources, Conservation and Recycling, 54(5), 303-309., 2010

122. Blumendorf, M. Building Sustainable Smart Homes. On Information and Communication Technologies, 2013

123. Dyllick, T., Hockerts, K. Beyond the Business Case for Corporate Sustainability. Business Strategy and the Environment (11:2), pp. 130-141., 2002

124. Cohen, S. M., Curd, P., Reeve, C. D. C. (Eds.). Readings in ancient Greek philosophy: from Thales to Aristotle. Hackett Publishing, 2005

125. Wooldridge, M., Jennings, N.R. Intelligent agents: theory and practice. The Knowledge Engineering Review, 1995

126. Sabharwal, A., Selman, B. S. Russell, P. Norvig. Artificial Intelligence: A Modern Approach., 2011

127. Franklin S., Graesser A. Is it an agent, or just a program: a taxonomy for autonomous agents. in: J.P. Muller, M.I. Wooldridge, N.R. Jennings (Eds.), Intelligent Agents 3, Lecture Notes in Artificial Intelligence, Vol. 1193, Springer, Berlin, 1997

128. C. Tessier, L. Chaudron, H.J. Muller, editors. Conflicting Agents: Conflict Management in Multi-Agent Systems. Kluwer Academic Publishers, 2002

129. Kraus S. Negotiation and Cooperation in Multi-Agent Environments. Artificial Intelligence 94, 79-97, 1997

130. Burmeister B., Haddadi A., Sundermeyer K. Generic, Configurable, Cooperation, Cooperation Protocols for Multi-Agent Systems. in: Castelfranchi C., Muller J.-P, editors From Reaction to Cognition (MAAMAW'93), LNAI 957, Springer Verlag, Berlin/Heidelberg/New York, Germany, pp. 157-171, 1995

131. Chavez A., Moukas A., Maes P. Challenger: A multi-agent system for distributed resource allocation. in Proc. Autonomous Agents, Marina del Rey, CA, pp.323-331, 1997

132. Shehory O., Sycara K., Chalasani P., Jha S. Agent cloning: an approach to agent mobility and resource allocation. IEEE communications Magazine 36(7), 58-67, 1998

133. Chau M., Zeng D., Chen H., Huang M., Hendriawan D. Design and evaluation of a multi-agent collaborative web mining system. Decision Support Systems, Special Issue on Web Retrieval and Mining, 35(1), 167-183, 2003

134. Cabri G., Leonardi L., Zambonelli F. Engineering mobile agent applications via context-dependent coordination. IEEE Trans. Softw. Eng. 28, 11, 1034-1051, 2002

135. Simon, H. A. Models of man; social and rational. 1957

136. Durfee, E. H., Lesser, V. R. Negotiating task decomposition and allocation using partial global planning. Distributed artificial intelligence, 2(1), 229-244., 1989

137. N.R. Jennings, K. Sycara, M. Woodridge. A roadmap of agent research and development, Autonomous Agents and MultiAgent Systems 1 (1) 7 – 38., 1999

138. Rao, A. S., Georgeff, M. P. BDI agents: From theory to practice. In ICMAS (Vol. 95, pp. 312-319)., 1995

139. Kinny, D., George, M. Commitment and Effectiveness of Situated Agents. In IJCAI-91 (pp. 82-88)., 1991

140. Bratman, M. Intention, plans, and practical reason. 1987

141. Bratman, M. E., Israel, D., Pollack, M. E. Plans and resource-bounded practical reasoning. Computational intelligence, 4(4), 349-355., 1988

142. Georgeff, M. P., Lansky, A. L. Reactive reasoning and planning. In AAAI (Vol. 87, pp. 677-682), 1987

143. Georgeff, M. P., Ingrand, F. F. Decision-making in an embedded reasoning system (pp. 972-978). Australian Artificial Intelligence Institute, 1989

144. Railsback, S. F., Grimm, V. Agent-based and individual-based modeling: a practical introduction. Princeton university press, 2011

145. Standish, R. K. On complexity and emergence. arXiv preprint nlin/0101006., 2001

146. Burmeister, B., Haddadi, A., Matylis, G. Application of multi-agent systems in traffic and transportation. In Software Engineering. IEE Proceedings-[see also Software, IEE Proceedings] (Vol. 144, No. 1, pp. 51-60). IET., 1997

147. Sharples, S., Callaghan, V., Clarke, G. A multi-agent architecture for intelligent building sensing and control. Sensor Review, 19(2), 135-140., 1999

148. Giladi, R. Heterogeneous Building Automation and IP Networks Management. ICDCS Workshops, pp. 636-641., 2004

149. Schatten, M. Smart Residential Buildings as Learning Agent Organizations in the Internet of Things, Business Systems Research, Vol. 5, No. 1. pp. 34-46, 2014

150. Watson, R. T., Boudreau, M. C., Chen, A. J. Information systems and environmentally sustainable development: energy informatics and new directions for the IS community. MIS quarterly, 34(1), 23-38., 2010

151. Rutishauser U., Schafer, A. Adaptive Building Automation. A multi-Agent approach. Research project, 2002

152. Georgakarakou, C. E., Economides, A. A. Agent technology applied to Intelligent Buildings. In Proceedings of the 10th WSEAS international conference on Computers (pp. 780-784). World Scientific and Engineering Academy and Society (WSEAS), 2006

153. Sycara, K. P. Multi-agent Systems. AI magazine, Vol.19, No. 2, Intelligent Agents Summer, 1998

154. Callaghan, V., Clarke, G., Pounds-Cornish, A., & Sharples, S. Buildings as intelligent autonomous systems: a model for integrating personal and building agents. In The 6th International Conference on Intelligent Autonomous Systems, IAS-6 pp. 410-415, 2000

155. Aha, D. W. Case-based learning algorithms. In Proceedings of the 1991 DARPA Case-Based Reasoning Workshop, Vol. 1, pp. 147-158, 1991

156. Cook, D. J., Youngblood, M., Heierman III, E. O., Gopalratnam, K., Rao, S., Litvin, A., & Khawaja, F. MavHome: An agent-based smart home. In Pervasive Computing and Communications, 2003. (PerCom 2003). Proceedings of the First IEEE International Conference on, pp. 521-524. IEEE, 2003

157. Widmer, R., Oswald-Krapf, H., Sinha-Khetriwal, D., Schnellmann, M., Böni, H. Global perspectives on e-waste. Environmental Impact Assessment Review, 25(5):436–458. Environmental and Social Impacts of Electronic Waste Recycling., 2005

158. Intille, S. S. Designing a home of the future. IEEE Pervasive Computing, 1(2):76–82, 2002

159. McCalley, L. T. From Motivation and Cognition Theories to Everyday Applications and Back Again: The Case of Product-Integrated Information and Feedback, Energy Policy (34:2), pp. 129-137., 2006

160. Wilson, C., and Dowlatabadi, H. Models of Decision Making and Residential Energy Use. Annual Review of Environmental Resources (32), pp. 169-203., 2007

161. Parker, D., Hoak, D., Meier, A., Brown, R. How much energy are we using? Potential of residential energy demand feedback devices. Proceedings of the 2006 Summer Study on Energy Efficiency in Buildings, American Council for an Energy Efficient Economy, Asilomar, CA, 2006

162. Froehlich, J., Findlater, L., Landay, J. The design of eco-feedback technology. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 1999-2008. ACM, 2010

163. Intille, S. S. The goal: smart people, not smart homes. In Proceedings of ICOST2006: The International Conference on Smart Homes and Health Telematics. Amsterdam: IOS Press, pp. 3-6, 2006

164. Mankoff, J., Matthews, D., Fussell, S. R., Johnson, M. Leveraging social networks to motivate individuals to reduce their ecological footprints. In System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on, pp. 87-87. IEEE, 2007

165. Yun, T. J. Investigating the impact of a minimalist in-home energy consumption display. In CHI'09 Extended Abstracts on Human Factors in Computing Systems, pp. 4417-4422. ACM, 2009

166. Costanza, E., Ramchurn, S. D., Jennings, N. R. Understanding domestic energy consumption through interactive visualisation: a field study. In Proceedings of the 2012 ACM Conference on Ubiquitous Computing, pp. 216-225. ACM, 2012

167. Abras, S., Pesty, S., Ploix, S., Jacomino, M. Advantages of MAS for the resolution of a power management problem in smart homes. In Advances in Practical Applications of Agents and Multiagent Systems, pp. 269-278. Springer Berlin Heidelberg, 2010

168. Palensky, P., Dietrich, D., Posta, R., Reiter, H. Demand Side Management in private homes by using LonWorks. In Factory Communication Systems, 1997. Proceedings. 1997 IEEE International Workshop on, pp. 341-347. IEEE, 1997

169. Karlgren, J., Fahlén, L. E., Wallberg, A., Hansson, P., Ståhl, O., Söderberg, J., Åkesson, K. P. Socially intelligent interfaces for increased energy awareness in the home. In The Internet of Things, pp. 263-275. Springer Berlin Heidelberg, 2008

170. Gadakari, T., Mushatat, S., and Newman, R. Can Intelligent Buildings Lead Us to a Sustainable Future? 3rd International Conference on Engineering, Project and Production Management. Brighton, UK 10-11 September 2012, pp.335-346., 2012

171. Lucidarme, P., Simonin, O., Li´ egeois, A. Implementation and evaluation of a satisfaction/altruism based architecture for multi-robot systems. In: Proceedings of the 2002 IEEE International Conference on Robotics and Automation, Washington, USA, August 1-5, pp.1007–1012, 2002

172. Ben-Ari, M. Principles of Concurrent and Distributed Programming. Prentice-Hall, Englewood Cliffs, NJ., 1990

173. Holzmann, G. Design and Validation of Computer Protocols. Prentice-Hall International, Heme1 Hempstead, UK, 1991

174. Magee, J., Kramer, J. Concurrency. John Wiley and Sons, Chichester, 1999

175. Binmore, K. Fun and Games: A Text on Game Theory. D. C. Heath and Company, 1992

176. Garey, M. R. Johnson, D. S. Computers and Intractability: a Guide to the Theory of NP-Completeness. W. H. Freeman, New York., 1979

177. Papadimitriou, C. H. Computational Complexity. Addison-Wesley, Reading, MA, 1994

178. Atzori, L., Iera, A., Morabito, G. The internet of things: a survey, Computer Networks: The International Journal of Computer and Telecommunications Networking, Vol. 54, No. 15, pp. 2787–2805., 2010

179. Hu, Z. et al. Iterative Model-based Identification of Building Components and Appliances by Means of Sensor-Actuator Networks, in Segovia, R., Zarli, A., Fies, B. (Eds.), Proceedings of the 2nd Workshop organised by the EEB Data Models Community "EEBuilding Data Models", 26-28 October 2011, Sophia Antipolis, France, Publications Office of the European Union: Luxembourg, pp. 216-226., 2011

180. Ayala, I., Amor, M., Fuentes, L. An agent platform for self-configuring agents in the internet of things. In *Third International Workshop on Infrastructures and Tools for Multiagent Systems, ITMAS* (pp. 65-78)., 2012

181. Vlacheas, P. et al. Enabling smart cities through a cognitive management framework for the internet of things, IEEE Communications Magazine, Vol. 51, No. 6, pp. 102-111, 2013

182. Su, K., Li, J., Fu, H. Smart city and the applications. In Electronics, Communications and Control (ICECC), 2011 International Conference on (pp. 1028-1031). IEEE., 2011

183. Komninos, N. The architecture of intelligent clities: Integrating human, collective and artificial intelligence to enhance knowledge and innovation. In Intelligent Environments, 2006. IE 06. 2nd IET International Conference on (Vol. 1, pp. 13-20). IET., 2006

184. Qin, H., Li, H., Zhao, X. Development status of domestic and foreign smart city. Global Presence, 9, 50-52., 2010

185. Batty, M., Axhausen, K. W., Giannotti, F., Pozdnoukhov, A., Bazzani, A., Wachowicz, M., ... Portugali, Y. Smart cities of the future. The European Physical Journal Special Topics, 214(1), 481-518, 2012

186. Yun, M., Yuxin, B. Research on the architecture and key technology of internet of things (IoT) applied on smart grid, in Tian, X. (Ed.), International Conference on Advances in Energy Engineering (ICAEE), 19-20 June 2010, Beijing, China, Institute of Electrical and Electronics Engineers: Piscataway, pp. 69–72., 2010

187. Bui, N., Zorzi, M. Health care applications: a solution based on the internet of things, in Fratassi, S., Marchetti, N. (Eds.), Proceedings of the 4th international symposium on applied sciences in biomedical and communication technologies, ACM: New York, Article no. 131., 2011

188. Vermesan, O. et al. Internet of things strategic research roadmap, in Vermesan, O., Friess, P. (Eds.), Internet of Things: Global Technological and Societal Trends, River Publisher, Aalborg, pp. 9–52., 2011

189. Welbourne, E. et al. Building the internet of things using rfid: the rfid ecosystem experience, Internet computing, Vol. 13, No. 3, pp. 48–55., 2009

190. Dohr, A. et al. The internet of things for ambient assisted living, in Latifi, S. (Ed.), Seventh International Conference on Information Technology: New Generations, 12-14 April, 2010, Las Vegas, USA, IEEE, Piscataway, pp. 804–809., 2010

191. Motik, B., Šimleša, D. Zeleni alati za održivu revoluciju. Retrieved February 15, 2015, from: http://www.zmag.hr/admin/public/javascript/fckeditor/editor/ckfinder/userfiles/files/zeleni-alati.pdf

192. Motik, B. Tehnologije za održivi svijet. Retrieved February 15, 2015, from: http://www.zmag.hr/admin/public/javascript/fckeditor/editor/ckfinder/userfiles/files/prirucnik%20tehnologije%20za%20odrzivi%20svijet.pdf

193. Motik, B. Solarni kolektori - priručnik za samogradnju. Retrieved February 15, 2015, from: http://www.zmag.hr/admin/public/javascript/fckeditor/editor/ckfinder/userfiles/files/prirucnik%20SK%20za%20web.pdf

194. Zoković, I., Motik, B., Rodik, D., Luketina, K. Zeleni alati - grijemo se i kuhamo suncem. Retrieved February 15, 2015, from: http://www.zmag.hr/admin/public/javascript/fckeditor/editor/ckfinder/userfiles/files/zeleni_alati_sunce_download.pdf

195. ZMAG: Kalendar tečajeva i radionica za 2015. Retrieved February 15, 2015, from http://www.zmag.hr/hr/radionice,-tecajevi,-dogadjanja

196. Motik, B., Rodik, D., Šimleša, D., Dragičević, G., Kardum I., Šišak, M. Maljković, N., Pocrnčić, S. ,Paro Vidolin S., Pešak S. Permakulturni dizajn - priručnik uz tečaj. Retrieved February 15, 2015, from http://www.zmag.hr/admin/public/javascript/fckeditor/editor/ckfinder/userfiles/files/PRIRUC%CC%8CNIK_PK_DIZAJN_web.pdf

197. PGE SmartAC program. Retrieved February 25, 2015, from: http://www.pge.com/en/myhome/saveenergymoney/plans/smartac/index.page

198. Balijepalli, Murthy; Pradhan, Khaparde. Review of Demand Response under Smart Grid Paradigm. IEEE PES Innovative Smart Grid Technologies, 2011

199. Albadi, M. H., El-Saadany E.F. Demand Response in Electricity Markets: An Overview. IEEE., 2007

200. Wooldridge, M., Jennings, N. R., Kinny, D. The Gaia methodology for agent-oriented analysis and design. Autonomous Agents and multi-agent systems, 3(3), 285-312., 2000

201. Mark, Sept. Rapid GUI programming with Python and Qt. Learner's Guide to PyQt Programming 2, 2009

202. Kuhlman, D. A Python Book: Beginning Python, Advanced Python, and Python Exercises. 2009

203. Oliphant, Travis E. Python for scientific computing. Computing in Science & Engineering 9, no. 3 (2007): 10-20., 2007

204. Millman, K. Jarrod, Aivazis M. Python for scientists and engineers. Computing in Science & Engineering 13, no. 2 (2011): 9-12., 2011

205. Mollison, B., &Holmgren, D. Permaculture one. Morebank, NSW Australia: Transworld Publications., 1978

206. Holmgren, D. Permaculture: Principles and pathways beyond sustainability. Holmgren Design Services, 2003

207. Holzer, S. Sepp Holzer's permaculture: a practical guide to small-scale, integrative farming and gardening. Chelsea Green Publishing, 2011

208. Hemenway, T. Gaia's garden: A guide to home-scale permaculture. White River Junction, Vermont: Chelsea Green Publishing Company, 2001

209. Bell, G. The permaculture way: Practical steps to create a self-sustaining world. Hampshire, United Kingdom: Permanent Publications, 2004

210. European Environment Agency. Retrieved February 25, 2015 from: http://www.eea.europa.eu/soer/countries/hr/soertopic_view?topic=freshwater

211. Climate data, Croatian Meteorological and Hydrological Service. Retrieved February 28, 2015 from: http://klima.hr/klima.php?id=k1&param=srednjak&Grad=sisak

212. Thomas, T. H., Martinson, D. B. Roofwater harvesting: a handbook for practitioners. In IRC Technical paper series (No. 49). IRC, 2007

213. Ghisi, Enedir, Andreza Montibeller, Schmidt R.W. Potential for potable water savings by using rainwater: An analysis over 62 cities in southern Brazil. Building and Environment 41, no. 2 (2006): 204-210., 2006

214. Aladenola, O. O., Adeboye, O. B. Assessing the potential for rainwater harvesting. Water Resources Management, 24(10), 2129-2137., 2010

215. Inocencio, Arlene B., Padilla J.E., Esmyra P. Javier. Determination of basic household water requirements. Philippine Institute for Development Studies, 1999

216. Connors, Mary M., Albert A. Harrison, and Faren R. Akins. Living aloft: Human requirements for extended spaceflight, 1985

217. Kortenkamp, David, R. Peter Bonasso, and Devika Subramanian. Distributed, autonomous control of space habitats. In Aerospace Conference, 2001, IEEE Proceedings., vol. 6, pp. 2751-2762. IEEE, 2001

218. Clarke, Graham, V. Callaghan, and A. Pounds-Cornish. Intelligent habitats and the future: the interaction of people, agents and environmental artefacts. In 4S/EASST Conference on Technoscience, Citizenship and Culture in the 21st Century, Vienna, pp. 26-28. 2000

219. Sharples S, Callaghan V, Clarke, G. A Self-Learning Multi-Agent Architecture for Intelligent Building Monitoring and Control", Int'l Sensor Review Journal, Vol. 19. No. 1, May 1999

220. Callaghan V, Clarke G, Colley M, Hagras H. A Soft-Computing DAI Architecture for Intelligent Buildings. Journal of Studies in Fuzziness and Soft Computing on Soft Computing Agents, Physica-Verlag-Springer, November 2000

221. Johnson, Richard D., and Holbrow C. Space Settlements. A Design Study. NASA, Washington, SP-413, 1976

222. Mars One, "Mars One - Will the astronauts have enough water, food and oxygen?". Retrieved March 25, 2015 from: http://www.mars-one.com/faq/health-and-ethics/will-the-astronauts-have-enough-water-food-and-oxygen

223. Do, Sydney, Koki Ho, Samuel Steven Schreiner, Andrew Charles Owens, and Olivier L. de Weck. An Independent Assessment of the Technical Feasibility of the Mars One Mission Plan., 2014

224. Mars One, "Mars One: Technical Feasibility". Retrieved March 25, 2015 from: http://www.mars-one.com/mission/technical-feasibility

225. Carter, Layne, Barry Tobias, and Nicole Orozco. "Status of ISS Water Management and Recovery." In 42nd International Conference on Environmental Systems, San Diego, CA, 15-19 July 2012, pp. 1-12. 2012

226. Torpor inducing transfer habitat for human stasis to Mars. Retrieved March 25, 2015 from: http://www.sei.aero/eng/papers/uploads/archive/NIAC-Torpor-Habitat-for-Human-Stasis-2-28-2014.pdf

227. Dumke, R., Mencke, S., Wille, C. Quality Assurance of Agent-Based and Self-Managed Systems. CRC Press, 2009

228. Raiffa, H. The art and science of negotiation. Harvard University Press, 1982

229. PVsyst photovoltaic software. Retrieved October 10, 2015 from: http://www.pvsyst.com/

230. Meteonorm: Irradiation data for every place on Earth. Retrieved October 10, 2015 from: http://meteonorm.com/

231. Wikipedia contributors. Nominal power (photovoltaic). Wikipedia, The Free Encyclopedia. Retrieved November 02, 2015 from: https://en.wikipedia.org/wiki/Nominal_power_(photovoltaic)

232. PVsyst 6.0-Photovoltaic system study (PVsyst 6 Help). Retrieved October 10, 2015 from: http://www.pvsyst.com/en/

233. Chevaleyre, Y., Dunne, P.E., Endriss, U., Lang, J., Lemaitre, M., Maudet, N., Padget, J., Phelps, S., Rodriguez-Aguilar, J.A., Sousa, P. Issues in multiagent resource allocation. *Informatica*, *30*(1), 2006

234. Shi B., Liu J. Self-organizing agents for efficient sustainable resource utilization. InProceedings of the The 2012 IEEE/WIC/ACM International Joint Conferences on Web

Intelligence and Intelligent Agent Technology-Volume 02 (pp. 510-517). IEEE Computer Society, 2012

235. La Rocca, G.L. Principles for Sustainable and Efficient Settlements, Based On A Multi-Agent Interface Between The Physical Spaces Structure And The Energy Grid. In Qatar Foundation Annual Research Conference, 2013

236. Sen, S. Tools for a robust, sustainable agent community. In Agents in Principle, Agents in Practice (pp. 2-2). Springer Berlin Heidelberg, 2011

237. Polaków, G., Metzger, M. Agent-based control system for sustainable wastewater treatment process. In Multi-disciplinary Trends in Artificial Intelligence (pp. 202-213). Springer Berlin Heidelberg, 2012

APPENDIX A Self-sustainability experts - short biographies

Consultant #1

Loren Amelang, Control Systems Engineer, Northern California, 35 years living off the grid full-time.

Loren built the solar smart house that uses computer monitoring to control and regulate house utilities such as solar, electric, water and ventilation. The south side of the house is covered in solar capture devices: solar hot water panels, a greenhouse, a solar hot air collector, and photovoltaic panels. He wrote about 10,000 lines of low-level device-specific code in order for home's water and electric systems to be operated more efficiently and automatically.

Current Skills (excerpt from the Consultant's CV):

"User interface and hands-on-hardware programming with C++ and C, particularly Keil C51 and C166 for SiLabs and Atmel, and CCS/MPLAB for Microchip; CAN and SAE J1939/ISObus; MS Visual Studio, .NET, MFC, and ActiveX for PC and WinMobile; Galil Motion Control; Idec/WIindLGC Smart Relay, HTML and scripting.

Computer system design, integration, administration and networking with Windows and MS-DOS, Linux, Android, Cisco, TCP/IP, motion control, vision/OCR, audio/video, touchscreen, Bluetooth, Wi-Fi, ZigBee.

PC board and electromechanical system design, integration, troubleshooting and repair, including PC, PLC, TTL, relay, and analog controls, servo motion, parallel/serial/network I/O, pneumatics, solar electric, solar thermal, and conventional power distribution and control.

Technical research, writing, documentation, standards compliance management, and meticulous technical proofreading. "

Consultant #2

Bruno Motik, permaculture activist, designer, builder and teacher; member of the Croatian Green Network of Activist Groups (Zelena mreža aktivističkih grupa, ZMAG). Author and co-author of several freely available books, handbooks, and tutorials, on topics such as sustainable revolution [191], sustainable technologies [192], water solar collectors [193], specific solar technologies [194].

Bruno is a founder of the “recycled estate” (eco-village) “Vukomerić”, which mainly serves as a test platform for the sustainable, off-grid technologies and permaculture design and principles, as well as an educational center for such topics, organizing numerous workshops and courses throughout the year [195]. The estate is geographically located in the vicinity of Zagreb, and it is the biggest ZMAG project so far. It is founded in the year 2000.

Consultant #3

Wished to remain anonymous; a founder of a local self-sustainable eco-village and a permaculture educator focused on the subjects of building with natural materials.

Internationally experienced in building human habitats using natural and locally available materials, both practical and theoretical, maintaining a wide international network with other professionals in the subject area.

Participated in and/or organized numerous seminars, workshops and courses covering topics of self-sustainability, construction, managing natural resources, etc. Co-author of the course handbook “Permaculture design”. [196].

APPENDIX B Implementation of Scenario Models

```
################### SCENARIO 1: PERMACULTURE ######################

# RESOURCE TRANSFER COSTS (defined by the user):
storage1Costs = {"UNIT-2" : 1, "UNIT-3" : 2, "UNIT-1-2" : 100}
storage2Costs = {"UNIT-1" : 1, "UNIT-3" : 1, "UNIT-1-2" : 100}
storage3Costs = {"UNIT-1" : 2, "UNIT-2" : 1, "UNIT-1-2" : 100}
storage12Costs = {"UNIT-1" : 100, "UNIT-2" : 100, "UNIT-3" : 100}

# NEGOTIATIONS PARAMETERS (defined by the user):
buyerValue = [0.99,-0.01]
buyerStrategy = [0,0.008]
sellerValue = [-0.6,0.4]
sellerStrategy = [1,-0.004]
negTimerMax = 1000
worth = 100000

# ------------------- HOUSE 1

 # storage parameters: name, crl, maxCapacity, lower threshold, upper
threshold, transfer costs, acceptable transfer cost value, resource ID

storage1 = StorageAgent("UNIT-1", 700, 2000, 300, 1700, storage1Costs, 20, 1,
buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)

 # collected rainfall for the September:
productionDistribution1 =
[76.5,855,0,103.5,85.5,0,13.5,0,0,0,0,0,751.5,1647,292.5,0,0,0,0,1309.5,0,0,0,
0,112.5,0,0,562.5,193.5,0,0]
producer1 = agent ("RAINFALL-U1", 1, productionDistribution1, 1, 30,
storages[0], 1)

 # residents collecting water with 20-liter canisters:
productionDistribution2 =
[60,0,40,0,0,40,60,60,60,60,60,60,0,0,0,60,60,60,60,0,60,40,20,0,60,60,0,60,60
,60]
```

```
producer2 = agent ("HANDWORK-U1", 1, productionDistribution2, 1, 30,
storages[0], 1)

 # 50 lit of water per human per day
consumptionDistribution1 = [-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-
50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50]
cunsumer1 = agent ("Person1-U1", 10, consumptionDistribution1, 1, 30,
storages[0], 0.5)
cunsumer2 = agent ("Person2-U1", 10, consumptionDistribution1, 1, 30,
storages[0], 0.5)
cunsumer3 = agent ("Person3-U1", 10, consumptionDistribution1, 1, 30,
storages[0], 0.5)
cunsumer4 = agent ("Person4-U1", 10, consumptionDistribution1, 1, 30,
storages[0], 0.5)
cunsumer5 = agent ("Person5-U1", 9, consumptionDistribution1, 1, 30,
storages[0], 0.6)
cunsumer6 = agent ("Person6-U1", 9, consumptionDistribution1, 1, 30,
storages[0], 0.6)

 # irrigation in the dry days, 10 l/m2, garden 50 m2 => 500 l
consumptionDistribution2 = [0,0,0,0,0,0,0,0,0,-500,0,-500,0,0,0,0,-
500,0,0,0,0,-500,0,0,0,0,0,0,0,0]
cunsumer7 = agent ("IRRIGATION-U1", 2, consumptionDistribution2, 1, 30,
storages[0], 0.2)

 # various (semi-random)
consumptionDistribution3 = [-2,0,0,-10,0,0,-4,-1,0,-12,0,-1,-1,0,-3,0,0,-
21,0,-1,0,-5,-2,-3,0,-7,0,-2,0,-1]
cunsumer8 = agent ("VA-U1", 1, consumptionDistribution3, 1, 30, storages[0],
0.1)



# ------------------- HOUSE 2

storage2 = StorageAgent("UNIT-2", 350, 1440, 250, 1100, storage2Costs, 20, 1,
buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)
```

```
 # collected rainfall for the September:
productionDistribution3 =
[63.75,712.5,0,86.25,71.25,0,11.25,0,0,0,0,0,626.25,1372.5,243.75,0,0,0,0,1091
.25,0,0,0,0,93.75,0,0,468.75,161.25,0,0]
producer3 = agent ("RAINFALL-U2", 1, productionDistribution3, 1, 30,
storages[1], 1)

 # manually collecting water with 20-liter canisters:
productionDistribution4 =
[0,0,40,0,0,40,20,0,0,20,40,0,0,20,40,40,40,0,0,0,20,0,40,0,40,20,0,20,20,0]
producer4 = agent ("HANDWORK-U2", 1, productionDistribution4, 1, 30,
storages[1], 1)

 # 50 lit of water per human per day
consumptionDistribution4 = [-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-
50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50]
cunsumer9 = agent ("Person1-U2", 10, consumptionDistribution4, 1, 30,
storages[1], 0.5)
cunsumer10 = agent ("Person2-U2", 9, consumptionDistribution4, 1, 30,
storages[1], 0.6)

 # irrigation in the dry days
consumptionDistribution5 = [0,0,0,0,0,0,0,0,0,0,-300,0,0,0,0,0,0,-
300,0,0,0,0,-300,0,0,0,0,0,0,0]
cunsumer11 = agent ("IRRIGATION-U2", 2, consumptionDistribution5, 1, 30,
storages[1], 0.2)

 # various (semi-random)
consumptionDistribution6 = [-1,-1,-2,0,-1,0,0,0,-1,-6,0,-1,-1,0,-2,0,0,-12,0,-
1,-1,-8,-2,-3,0,-7,0,-2,-5,-1]
cunsumer12 = agent ("VA-U2", 1, consumptionDistribution6, 1, 30, storages[1],
0.1)



# ------------------- HOUSE 3

storage3 = StorageAgent("UNIT-3", 700, 1500, 200, 800, storage3Costs, 20, 1,
buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)
```

```
 # collected rainfall for the September:
productionDistribution5 =
[91.8,1026,0,124.2,102.6,0,16.2,0,0,0,0,0,901.8,1976.4,351,0,0,0,0,1571.4,0,0,
0,0,135,0,0,675,232.2,0,0]
producer5 = agent ("RAINFALL-U3", 1, productionDistribution5, 1, 30,
storages[2], 1)

 # manually collecting water with 20-liter canisters:
productionDistribution6 =
[0,0,20,0,0,0,20,0,0,0,20,0,0,0,0,0,0,0,40,0,0,40,0,0,0,0,0,20,20,0]
producer6 = agent ("HANDWORK-U3", 1, productionDistribution6, 1, 30,
storages[2], 1)

 # 50 lit of water per human per day
consumptionDistribution7 = [-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-
50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50,-50]
cunsumer13 = agent ("Person1-U3", 10, consumptionDistribution7, 1, 30,
storages[2], 0.5)
cunsumer14 = agent ("Person2-U3", 10, consumptionDistribution7, 1, 30,
storages[2], 0.5)

 # irrigation in the dry days
consumptionDistribution8 = [0,0,0,0,0,0,0,0,0,-300,0,-300,0,0,0,0,-300,0,-
300,0,0,0,-300,0,0,0,0,0,0,0]
cunsumer15 = agent ("IRRIGATION-U3", 2, consumptionDistribution8, 1, 30,
storages[2], 0.3)

 # various (semi-random)
consumptionDistribution9 = [-7,-2,0,-1,0,-23,0,-2,-5,-1,0,0,0,0,0,-36,-2,-
21,0,0,0,0,0,0,-6,-2,0,-1,0,0]
cunsumer16 = agent ("VA-U3", 1, consumptionDistribution9, 1, 30, storages[2],
0.1)


storageEXP = StorageAgent("UNIT-1-2", 31, 500, 0, 500, storage12Costs, 20, 2,
buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)
```

```
consumptionDistributionEXP1 = [-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1]
cunsumerEXP1 = agent ("EXPERIMENTAL-CONSUMER", 1, consumptionDistributionEXP1,
1, 30, storages[3], 1)

productionDistributionEXP1 =
[2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2]
producerEXP1 = agent ("EXPERIMENTAL-PRODUCER", 1, productionDistributionEXP1,
1, 30, storages[3], 1)
```

Listing 12. A model of the scenario 1 (permaculture)

```
######################### SCENARIO 2: ENERGY PRODUCTION ######################


# RESOURCE TRANSFER COSTS (defined by the user):
storage1Costs = {"UNIT-2-PV" : 1, "UNIT-3-PV" : 2, "UNIT-1-Propane" : 100,
"UNIT-3-Propane" : 100}
storage2Costs = {"UNIT-1-PV" : 1, "UNIT-3-PV" : 1, "UNIT-1-Propane" : 100,
"UNIT-3-Propane" : 100}
storage3Costs = {"UNIT-1-PV" : 2, "UNIT-2-PV" : 1, "UNIT-1-Propane" : 100,
"UNIT-3-Propane" : 100}
storage12Costs = {"UNIT-1-PV" : 100, "UNIT-2-PV" : 100, "UNIT-3-PV" : 100,
"UNIT-3-Propane" : 100}
storage31Costs = {"UNIT-1-PV" : 100, "UNIT-2-PV" : 100, "UNIT-3-PV" : 100,
"UNIT-1-Propane" : 100}

# NEGOTIATIONS PARAMETERS (defined by the user):
buyerValue = [0.99,-0.01]
buyerStrategy = [0,0.09]
sellerValue = [-0.6,0.4]
sellerStrategy = [1,-0.004]
negTimerMax = 1000
worth = 100000


# ------------------- UNIT 1

# PV array
storage1 = StorageAgent("UNIT-1-PV", 5700, 7080, 2000, 6500, storage1Costs,
20, 1, buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax,
worth)

#AUX
storage12 = StorageAgent("UNIT-1-Propane", 10, 20, -1, 20, storage12Costs, 20,
2, buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)

#Propane
consumptionDistributionP = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,-1,-2,0,-
7,0,0,0,0,0,0,0,0,0,0,0]
cunsumerP = agent ("Propane-U1", 10, consumptionDistributionP, 1, 30,
storages[1], 1)


 # PV array:
productionDistribution1 =
[2970,2565,2236,1277,1149,1080,1578,1757,2100,1513,1208,1255,1005,1001,2841,25
49,2684,2605,2156,2780,504,731,573,432,688,1987,3291,3428,2888,900]
producer1 = agent ("PV-U1", 1, productionDistribution1, 1, 30, storages[0], 1)

 # AUX:
productionDistribution2 =
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,120,240,0,840,0,0,0,0,0,0,0,0,0,0,0]
producer2 = agent ("AUX-U1", 1, productionDistribution2, 1, 30, storages[0],
1)

 # Illumination
```

```
consumptionDistribution1 = [-138,-83,-242,-192,-141,-154,-126,-202,-169,-261,-
231,-169,-280,-113,-247,-274,-132,-220,-261,-267,-137,-227,-197,-273,-219,-
165,-117,-175,-278,-134]
cunsumer1 = agent ("Illumination-U1", 10, consumptionDistribution1, 1, 30,
storages[0], 0.5)

 # Consumer electronics
consumptionDistribution2 = [-289,-187,-487,-437,-586,-291,-470,-256,-561,-
856,-735,-730,-610,-575,-494,-950,-775,-432,-396,-418,-575,-855,-307,-369,-
381,-916,-877,-270,-437,-356]
cunsumer2 = agent ("CEs-U1", 2, consumptionDistribution2, 1, 30, storages[0],
0.5)

 # Refrigerator
consumptionDistribution3 = [-724,-722,-729,-722,-708,-759,-716,-709,-774,-
734,-795,-745,-738,-737,-739,-757,-732,-783,-799,-732,-756,-798,-740,-719,-
743,-763,-752,-777,-779,-755]
cunsumer3 = agent ("Fridge-U1", 1, consumptionDistribution3, 1, 30,
storages[0], 1)

 # Washing Machine
consumptionDistribution4 = [-0,-0,-0,-0,-0,-0,-0,-0,-0,-0,-0,-0,-1060,-0,-0,-
1080,-0,-0,-1100,-0,-0,-1040,-0,-0,-0,-0,-0,-1060,-0,-0]
cunsumer4 = agent ("Washer-U1", 1, consumptionDistribution4, 1, 30,
storages[0], 0.7)

 # Tools
consumptionDistribution5 = [-0,-256,-0,-0,-0,-0,-185,-0,-750,-620,-0,-732,-
1635,-0,-326,-756,-1030,-415,-650,-0,-0,-0,-0,-0,-0,-0,-0,-0,-0,-0]
cunsumer5 = agent ("Tools-U1", 1, consumptionDistribution5, 1, 30,
storages[0], 0.7)

 # Inverter
consumptionDistribution6 = [-115.1,-124.8,-145.8,-135.1,-143.5,-120.4,-149.7,-
116.7,-225.4,-247.1,-176.1,-237.6,-432.3,-142.5,-180.6,-381.7,-266.9,-185,-
320.6,-141.7,-146.8,-292,-124.4,-136.1,-134.3,-184.4,-174.6,-228.2,-149.4,-
124.5]
cunsumer6 = agent ("Inverter-U1", 1, consumptionDistribution6, 1, 30,
storages[0], 1)


# ------------------- UNIT 2

storage2 = StorageAgent("UNIT-2-PV", 4200, 6120, 2500, 5000, storage2Costs,
20, 1, buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax,
worth)

 # PV array:
productionDistribution3 =
[2191,2031,1703,1127,00,1267,00,1108,00,1694,1811,2143,2191,1082,1270,1266,119
2,1481,2135,1368,2250,2072,1171,382,424,194,478,283,1505,1722,2148,1725,996]
producer3 = agent ("PV-U2", 1, productionDistribution3, 1, 30, storages[2], 1)

 # AUX:
productionDistribution4 =
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
```

```
producer4 = agent ("AUX-U2", 1, productionDistribution4, 1, 30, storages[2],
1)

 # Illumination
consumptionDistribution7 = [-268,-295,-257,-220,-276,-287,-248,-284,-272,-
233,-245,-243,-245,-227,-290,-245,-272,-256,-204,-270,-298,-288,-293,-208,-
270,-215,-237,-254,-232,-244]
cunsumer7 = agent ("Illumination-U2", 10, consumptionDistribution7, 1, 30,
storages[2], 0.6)

 # Consumer electronics
consumptionDistribution8 = [-232,-210,-327,-235,-212,-211,-350,-248,-309,-
318,-319,-209,-342,-318,-353,-357,-344,-250,-359,-283,-345,-294,-239,-234,-
278,-280,-260,-268,-321,-247]
cunsumer8 = agent ("CEs-U2", 5, consumptionDistribution8, 1, 30, storages[2],
0.5)

 # Refrigerator
consumptionDistribution9 = [-775,-724,-773,-736,-772,-739,-731,-736,-728,-
724,-702,-734,-734,-731,-711,-779,-752,-744,-728,-730,-773,-770,-766,-732,-
736,-753,-753,-767,-772,-719]
cunsumer9 = agent ("Fridge-U2", 1, consumptionDistribution9, 1, 30,
storages[2], 1)

 # Hydronic Pump
consumptionDistribution10 = [-63,-65,-88,-61,-44,-43,-86,-50,-99,-54,-58,-51,-
93,-63,-56,-47,-50,-63,-51,-49,-77,-41,-68,-100,-77,-82,-65,-57,-82,-83]
cunsumer10 = agent ("Pump-U2", 1, consumptionDistribution10, 1, 30,
storages[2], 1)

 # Tools
consumptionDistribution11 = [-0,-0,-0,-0,-0,-0,-0,-365,-120,-251,-0,-348,-
523,-0,-0,-0,-0,-0,-236,-0,-0,-0,-0,-0,-0,-0,-0,-0,-0,-0]
cunsumer11 = agent ("Tools-U2", 1, consumptionDistribution11, 1, 30,
storages[2], 0.7)

 # Inverter
consumptionDistribution12 = [-133.8,-129.4,-144.5,-125.2,-130.4,-128,-141.5,-
168.3,-152.8,-158,-132.4,-158.5,-193.7,-133.9,-141,-142.8,-141.8,-131.3,-
157.8,-133.2,-149.3,-139.3,-136.6,-127.4,-136.1,-133,-131.5,-134.6,-140.7,-
129.3]
cunsumer12 = agent ("Inverter-U2", 1, consumptionDistribution12, 1, 30,
storages[2], 1)


# ------------------- UNIT 3

# PV array
storage3 = StorageAgent("UNIT-3-PV", 680, 1008, 600, 800, storage3Costs, 20,
1, buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)

# AUX propane generator (dcl)
storage31 = StorageAgent("UNIT-3-Propane", 5, 10, -1, 20, storage31Costs, 20,
2, buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)

consumptionDistributionP3 = [0,0,0,0,0,0,0,0,0,0,0,0,-2,-2,0,0,0,0,0,0,0,0,0,-
1,0,0,0,0,0,0,0]
```

```
cunsumerP3 = agent ("propane-U3", 10, consumptionDistributionP3, 1, 30,
storages[4], 1)

 # PV array:
productionDistribution5 =
[261,450,412,64,70,41,414,438,473,295,70,124,126,75,370,321,344,402,326,307,21
,18,35,22,23,399,339,414,362,240]
producer5 = agent ("PV-U3", 1, productionDistribution5, 1, 30, storages[3], 1)

 # AUX:
productionDistribution6 =
[0,0,0,0,0,0,0,0,0,0,0,0,240,240,0,0,0,0,0,0,0,0,120,0,0,0,0,0,0,0]
producer6 = agent ("AUX-U1", 1, productionDistribution6, 1, 30, storages[3],
1)

 # Illumination
consumptionDistribution13 = [-88,-126,-86,-96,-160,-156,-137,-83,-151,-94,-
145,-126,-156,-86,-157,-99,-87,-119,-134,-138,-141,-150,-98,-112,-101,-105,-
132,-102,-127,-88]
cunsumer13 = agent ("Illumination-U3", 10, consumptionDistribution13, 1, 30,
storages[3], 0.6)

 # Consumer electronics
consumptionDistribution14 = [-137,-96,-170,-92,-71,-197,-153,-155,-133,-167,-
180,-105,-118,-169,-139,-89,-138,-79,-124,-72,-200,-82,-136,-69,-141,-96,-
179,-124,-85,-91]
cunsumer14 = agent ("CEs-U3", 2, consumptionDistribution14, 1, 30,
storages[3], 0.7)

 # Refrigerator
consumptionDistribution15 =
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
cunsumer15 = agent ("Fridge-U3", 1, consumptionDistribution15, 1, 30,
storages[3], 1)

 # Washing Machine
consumptionDistribution16 =
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
cunsumer16 = agent ("Washer-U3", 1, consumptionDistribution16, 1, 30,
storages[3], 0.7)

 # Tools
consumptionDistribution17 = [-0,-0,-0,-0,-0,-0,-0,-0,-0,-0,-0,-0,-351,-0,-0,-
0,-0,-0,-0,-0,-0,-0,-0,-0,-0,-0,-0,-0,-0,-0]
cunsumer17 = agent ("Tools-U3", 1, consumptionDistribution17, 1, 30,
storages[3], 0.8)

 # Inverter
consumptionDistribution18 = [-22.5,-22.2,-25.6,-18.8,-23.1,-35.3,-29,-23.8,-
28.4,-26.1,-32.5,-23.1,-62.5,-25.5,-29.6,-18.8,-22.5,-19.8,-25.8,-21,-34.1,-
23.2,-23.4,-18.1,-24.2,-20.1,-31.1,-22.6,-21.2,-17.9]
cunsumer18 = agent ("Inverter-U3", 1, consumptionDistribution18, 1, 30,
storages[3], 1)
```

Listing 13. A model of the scenario 2 (energy production)

```
###################### SCENARIO 3, pt.1: SPACE COLONY ##########################

# NEGOTIATIONS PARAMETERS (defined by the user):
buyerValue = [0.99,-0.01]
buyerStrategy = [0,0.008]
sellerValue = [-0.6,0.4]
sellerStrategy = [1,-0.004]
negTimerMax = 1000
worth = 100000

# RESOURCE TRANSFER COSTS (defined by the user):
storage1Costs = {"UNIT-2" : 1, "UNIT-3" : 2, "UNIT-4" : 3}
storage2Costs = {"UNIT-1" : 1, "UNIT-3" : 1, "UNIT-4" : 2}
storage3Costs = {"UNIT-1" : 2, "UNIT-2" : 1, "UNIT-4" : 1}
storage4Costs = {"UNIT-1" : 3, "UNIT-2" : 2, "UNIT-3" : 1}

# ------------------- UNIT 1

# storage parameters: name, crl, maxCapacity, lower threshold, upper
threshold, transfer costs, acceptable transfer cost value
# producer/consumer parameters: name, priority, capacity, workStart, workStop,
belongsTo, eFactor)

storage1 = StorageAgent("UNIT-1", 2000, 5000, 800, 4000, storage1Costs, 20, 1,
buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)

P1 =
[1245,1185,1275,1260,1140,1140,1125,1245,1140,1140,1275,1200,1260,1275,1125,11
40,1185,1155,1125,1275,1215,1230,1230,1185,1200,1215,1230,1230,1155,1275,1140,
1185,1215,1260,1215,1185,1170,1200,1230,1140,1185,1200,1275,1200,1275,1230,126
0,1140,1170,1260,1260,1185]
P1_WRS = agent ("Producer-WRS-U1", 1, P1, 1, 30, storages[0], 1)

P2 =
[257,238,154,293,195,212,271,210,213,182,235,278,184,280,345,276,310,323,238,1
71,170,374,171,251,270,190,355,330,279,234,206,316,240,266,224,278,214,173,207
,267,156,189,302,245,303,169,160,237,370,337,330,260]
P2_ISRU = agent ("Producer-ISRU-U1", 1, P2, 1, 30, storages[0], 1)

C1 = [-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-
1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-
1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-
1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-
1500,-1500]
C1_HUMAN = agent ("Consumer-HUMAN-U1", 1, C1, 1, 30, storages[0], 0.56)


# ------------------- UNIT 2
```

```
# storage parameters: name, crl, maxCapacity, lower threshold, upper threshold
storage2 = StorageAgent("UNIT-2", 2000, 5000, 800, 4000, storage2Costs, 20, 2,
buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)

P3 =
[1215,1155,1245,1260,1275,1140,1155,1155,1125,1260,1155,1215,1260,1245,1140,12
00,1155,1215,1170,1170,1230,1140,1260,1260,1230,1170,1140,1170,1155,1215,1230,
1170,1200,1275,1230,1215,1170,1125,1200,1140,1155,1200,1155,1140,1125,1185,118
5,1275,1275,1125,1245,1125]
P3_WRS = agent ("Producer-WRS-U2", 1, P3, 1, 30, storages[1], 1)

P4 =
[156,268,355,323,360,237,273,337,357,357,235,377,357,185,327,237,193,151,220,3
66,380,320,276,353,166,156,231,188,272,224,222,291,230,156,156,335,276,178,191
,321,285,229,301,241,266,202,312,355,237,320,356,158]
P4_ISRU = agent ("Producer-ISRU-U2", 1, P4, 1, 30, storages[1], 1)

C2 = [-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-
1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-
1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-
1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-1500,-
1500,-1500]
C2_HUMAN = agent ("Consumer-HUMAN-U2", 1, C2, 1, 30, storages[1], 0.56)


# ------------------- UNIT 3

# storage parameters: name, crl, maxCapacity, lower threshold, upper threshold
storage3 = StorageAgent("UNIT-3", 5000, 10000, 1000, 8000, storage3Costs, 20,
3, buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)

P5 =
[19170,21600,20250,22410,19170,22140,19710,21060,19440,21600,22950,20520,19170
,19440,20790,21060,18900,19440,22410,21330,19440,20250,21330,18900,21060,20520
,20250,19710,21870,20250,18900,19170,21330,22140,22410,21060,22950,19440,20790
,22950,20790,21870,21330,22680,21600,19710,21870,21600,18900,20520,21870,21870
]
P5_WRS = agent ("Producer-WRS-U3", 1, P5, 1, 30, storages[2], 1)

P6 =
[6899,6483,5725,5959,6455,5399,5633,5243,5445,6963,5523,5003,5212,5305,6089,59
55,6375,6733,5541,6837,5674,6464,6175,6670,6669,5409,6161,5212,6567,5203,6471,
6806,6186,5411,5322,5088,5228,5287,6479,6046,6650,5338,6501,5137,6685,6799,666
0,5967,5833,5013,5715,5294]
P6_ISRU = agent ("Producer-ISRU-U3", 1, P6, 1, 30, storages[2], 1)

C3 = [-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-
27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-
27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-
```

```
27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-
27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000]
C3_CROPS = agent ("Consumer-CROPS-U3", 1, C3, 1, 30, storages[2], 0.56)


# ------------------- UNIT 4

# storage parameters: name, crl, maxCapacity, lower threshold, upper threshold
storage4 = StorageAgent("UNIT-4", 5000, 10000, 1000, 8000, storage4Costs, 20,
4, buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)

P7 =
[21330,21600,22140,20250,20790,19980,22140,19710,21870,20790,20790,19440,19980
,19980,22410,21600,21060,21060,19440,21600,19980,21600,21060,19440,22410,19440
,21330,21330,21330,22680,21330,22140,18900,21060,21060,21060,22410,21330,20520
,19440,22410,22410,22410,22410,18900,20790,21600,21330,21600,21060,22140,18900
]
P7_WRS = agent ("Producer-WRS-U4", 1, P7, 1, 30, storages[3], 1)

P8 =
[6417,6236,5746,5905,6413,5446,5357,5090,5745,6577,5386,6241,6545,6572,5252,59
25,5157,6144,6202,5264,5101,5438,6182,6491,6481,5289,6585,5870,6743,6619,6345,
6210,6330,5329,6257,5748,5470,6204,6916,6151,6317,6865,6169,5452,6080,6897,500
4,6802,5707,6775,5612,5678]
P8_ISRU = agent ("Producer-ISRU-U4", 1, P8, 1, 30, storages[3], 1)

C4 = [-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-
27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-
27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-
27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-
27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000,-27000]
C4_CROPS = agent ("Consumer-CROPS-U4", 1, C4, 1, 30, storages[3], 0.56)
```

Listing 14. A model of the scenario 2 (space colony)

```
#################### SCENARIO 3, pt.2: SPACE FLIGHT ########################


# NEGOTIATIONS PARAMETERS (defined by the user):
buyerValue = [0.99,-0.01]
buyerStrategy = [0,0.008]
sellerValue = [-0.6,0.4]
sellerStrategy = [1,-0.004]
negTimerMax = 1000
worth = 100000

# RESOURCE TRANSFER COSTS (defined by the user):
storage1Costs = {"SS2" : 1, "SS3" : 2, "SS4" : 3, "SS5" : 4}
storage2Costs = {"SS1" : 1, "SS3" : 1, "SS4" : 2, "SS5" : 3}
storage3Costs = {"SS1" : 2, "SS2" : 1, "SS4" : 1, "SS5" : 2}
storage4Costs = {"SS1" : 3, "SS2" : 2, "SS3" : 1, "SS5" : 1}
storage5Costs = {"SS1" : 3, "SS2" : 2, "SS3" : 1, "SS4" : 1}

storage1 = StorageAgent("SS1", 1000, 1200, 200, 1000, storage1Costs, 20, 1,
buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)

P1 =
[924,948,948,1008,936,960,960,1020,936,1008,656,616,608,616,624,632,672,632,63
2,680,656,656,680,608,680,632,648,640,656,664,632,648,608,648,640,972,972,960,
936,924,948,972,996,960,1020]
P1_WRS = agent ("Producer-SS1-WRS", 1, P1, 1, 30, storages[0], 1)

P2 =
[166,196,163,207,124,176,143,140,115,174,171,196,217,103,152,205,151,164,171,1
76,111,123,189,187,190,179,174,117,175,117,136,151,196,120,156,125,160,175,200
,110,163,157,130,137,139]
P2_InputX = agent ("Producer-SS1-InputX", 1, P2, 1, 30, storages[0], 1)

C1 = [-1200,-1200,-1200,-1200,-1200,-1200,-1200,-1200,-1200,-
1200,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]
C1_crew = agent ("Crew1-SS1", 1, C1, 1, 10, storages[0], 0.56)

C2 = [0,0,0,0,0,0,0,0,0,0,-800,-800,-800,-800,-800,-800,-800,-800,-800,-800,-
800,-800,-800,-800,-800,-800,-800,-800,-800,-800,-800,-800,-800,-800,-
800,0,0,0,0,0,0,0,0,0,0]
C2_crew = agent ("Crew2-SS1", 1, C2, 11, 35, storages[0], 0.56)

C3 = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,-
1200,-1200,-1200,-1200,-1200,-1200,-1200,-1200,-1200,-1200]
C3_crew = agent ("Crew3-SS1", 1, C3, 36, 45, storages[0], 0.56)
```

```
storage2 = StorageAgent("SS2", 400, 1000, 150, 800, storage2Costs, 20, 1,
buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)

P3 =
[640,600,640,624,680,640,316,340,320,316,336,300,328,336,316,328,336,300,324,3
08,328,328,312,336,300,304,332,336,328,316,320,336,312,328,332,332,300,308,324
,616,632,680,656,600,648]
P3_WRS = agent ("Producer-SS2-WRS", 1, P3, 1, 30, storages[1], 1)

P4 =
[110,63,117,70,79,68,102,106,104,76,99,117,111,88,91,107,119,69,90,103,77,80,7
2,109,95,94,93,71,101,91,72,66,120,94,90,120,97,69,111,80,96,95,76,83,106]
P4_InputX = agent ("Producer-SS2-InputX", 1, P4, 1, 30, storages[1], 1)

C4 = [-800,-800,-800,-800,-800,-
800,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0]
C4_crew = agent ("Crew1-SS2", 1, C4, 1, 6, storages[1], 0.56)

C5 = [0,0,0,0,0,0,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-
400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-
400,-400,-400,-400,-400,-400,-400,0,0,0,0,0,0]
C5_crew = agent ("Crew2-SS2", 1, C5, 7, 39, storages[1], 0.56)

C6 =
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,-800,-800,-800,-800,-800,-800]
C6_crew = agent ("Crew3-SS2", 1, C6, 40, 45, storages[1], 0.56)


storage3 = StorageAgent("SS3", 400, 1000, 150, 800, storage3Costs, 20, 1,
buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)

P5 =
[672,616,672,648,632,640,332,328,336,304,308,304,300,340,308,336,340,316,328,3
16,300,328,312,300,336,324,308,312,324,308,336,300,308,316,332,316,312,320,312
,624,608,648,608,648,640]
P5_WRS = agent ("Producer-SS3-WRS", 1, P5, 1, 30, storages[2], 1)

P6 =
[88,93,94,72,82,85,97,69,110,75,73,77,70,82,98,105,89,82,111,110,85,62,89,64,9
2,117,106,98,72,78,64,66,72,69,90,98,75,80,93,83,65,100,79,108,95]
P6_InputX = agent ("Producer-SS3-InputX", 1, P6, 1, 30, storages[2], 1)

C7 = [-800,-800,-800,-800,-800,-
800,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0]
C7_crew = agent ("Crew1-SS3", 1, C7, 1, 6, storages[2], 0.56)
```

```
C8 = [0,0,0,0,0,0,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-
400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-
400,-400,-400,-400,-400,-400,-400,0,0,0,0,0,0]
C8_crew = agent ("Crew2-SS3", 1, C8, 7, 39, storages[2], 0.56)

C9 =
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,-800,-800,-800,-800,-800,-800]
C9_crew = agent ("Crew3-SS3", 1, C9, 40, 45, storages[2], 0.56)



storage4 = StorageAgent("SS4", 400, 1000, 150, 800, storage4Costs, 20, 1,
buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)

P7 =
[304,320,316,316,328,340,336,308,304,304,316,308,340,332,332,312,328,332,340,3
08,308,332,340,332,316,320,316,312,340,332,332,308,308,328,300,336,336,336,316
,340,320,300,324,316,308]
P7_WRS = agent ("Producer-SS4-WRS", 1, P7, 1, 30, storages[3], 1)

P8 =
[73,88,60,73,42,50,84,85,78,71,85,97,52,92,63,67,68,40,50,49,62,73,44,95,43,98
,97,72,53,100,42,63,95,96,99,49,65,91,73,99,85,82,61,83,51]
P8_InputX = agent ("Producer-SS4-InputX", 1, P8, 1, 30, storages[3], 1)

C10 = [-400,-400,-400,-400,-400,-
400,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0]
C10_crew = agent ("Crew1-SS4", 1, C10, 1, 6, storages[3], 0.56)

C11 = [0,0,0,0,0,0,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-
400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-
400,-400,-400,-400,-400,-400,-400,0,0,0,0,0,0]
C11_crew = agent ("Crew2-SS4", 1, C11, 7, 39, storages[3], 0.56)

C12 =
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,-400,-400,-400,-400,-400,-400]
C12_crew = agent ("Crew3-SS4", 1, C12, 40, 45, storages[3], 0.56)


storage5 = StorageAgent("SS5", 400, 1000, 150, 800, storage5Costs, 20, 1,
buyerValue, buyerStrategy, sellerValue, sellerStrategy, negTimerMax, worth)

P9 =
[316,304,316,340,340,332,328,304,324,324,320,340,308,316,304,316,320,324,316,3
40,324,308,340,308,316,316,340,340,328,316,340,316,312,340,312,340,336,312,300
,304,328,316,312,336,308]
```

```
P9_WRS = agent ("Producer-SS5-WRS", 1, P9, 1, 30, storages[4], 1)

P10 =
[49,67,99,59,62,66,60,83,90,50,60,44,92,82,49,84,76,55,95,89,85,84,46,42,87,92
,51,96,74,43,59,77,47,76,72,56,78,55,42,98,47,79,75,96,47]
P10_InputX = agent ("Producer-SS5-InputX", 1, P10, 1, 30, storages[4], 1)

C10 = [-400,-400,-400,-400,-400,-
400,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0]
C10_crew = agent ("Crew1-SS5", 1, C10, 1, 6, storages[4], 0.56)

C11 = [0,0,0,0,0,0,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-
400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-400,-
400,-400,-400,-400,-400,-400,-400,0,0,0,0,0,0]
C11_crew = agent ("Crew2-SS5", 1, C11, 7, 39, storages[4], 0.56)

C12 =
[0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,-400,-400,-400,-400,-400,-400]
C12_crew = agent ("Crew3-SS5", 1, C12, 40, 45, storages[4], 0.56)
```

Listing 15. A model of the scenario 3 (space flight)

Curriculum Vitae

Igor Tomičić was born on the May 5th 1983 in Karlovac. He finished the elementary school in Duga Resa city, and the high school in Karlovac. In 2007 He graduated “Information Systems” study programme on the Faculty of Organization and Informatics, University of Zagreb.

From 2007–2008 he worked in the private company NTH d.o.o. on operational positions involving digital telephony, VOIP protocols, linux servers, database maintenance, networking protocols, and Java services implementations in Debian environment. In the same time period he successfully mastered the Linux administration programme.

From the year 2008, he worked on the Faculty of Organization and Informatics as a junior scientist. In the same year in October, he enrolled the doctoral study “Information Sciences” on the same Faculty. He performed lectures in several courses, namely Information technology in managing complex systems, Multimedia systems, Digital image and text processing, Computer networks and advanced computer networks. He significantly contributed to the development of these courses, and managed operational transitions of some of them to open source platforms. He was student’s mentor on over 20 student undergraduate final papers.

An additional contribution to the professional training of the candidate was achieved by several Microsoft certificates (Security, Windows Server, Networking, Database Administration Fundamentals), summer school “NATO Engineering Dependable Software Systems”, and autumn school “COST IC0904 TwinTide Autumn Training School on Research Methods for Human-Computer Interaction”.

Privately, he is an active member of the local sustainable development association group, and the founder of the Community supported agriculture group covering the Varaždin county area.

List of Publications

Book Chapters

[1] Barbir, Saša; Bosilj, Neven; Dobrinić, Damir; Gerić, Sandro; Gregurec, Iva; Hutinski, Željko; Kirinić, Valentina; Mandić, Miroslav; Mekovec, Renata; Miličić, Miroslav; Rabuzin, Kornelije; Tomičić, Igor; Vranešević, Tihomir: Marketing i baze podataka. Dobrinić, Damir (ur.). Varaždin, Fakultet organizacije i informatike, 2011.

Journal papers

[1] Schatten, Markus; Ševa, Jurica; Tomičić, Igor: A Roadmap for Scalable Agent Organizations in the Internet of Everything. Journal of systems and software. 115 (2016); 31-41

[2] Tomičić, Igor; Schatten, Markus: Agent-based Framework for Modelling and Simulation of Resources in Self- Sustainable Human Settlements - A Case Study on Water Management in an Eco-Village Community in Croatia. International journal of sustainable development and world ecology. (2016)

[3] Tomičić, Igor; Turk, Matija; Lovrenčić, Mišo: A performance comparison of three SIP softswitches: Asterisk, FreeSWITCH, and Yate. Zbornik radova Međimurskog veleučilišta u Čakovcu. 6 (2016), 2; 147-157

[4] Tomičić, Igor; Schatten, Markus: Towards an Agent Based Framework for Modelling Smart Self-Sustainable Systems. Interdisciplinary description of complex systems. 13 (2015), 1; 50-63

[5] Tomičić, Igor; Ćorić, Ana; Klačmer Čalopa, Marina: Croatian banking sector research: relationship between ownership structure, concentration, owners' type and bank performance. Journal of Information and Organizational Sciences. 36 (2012), 2; 159-167

[6] Viktorovych Shkarupylo, Vadym; Tomičić, Igor; Mykolaiovych Kasian, Kostiantyn: The Investigation of TLC Model Checker Properties. Journal of Information and Organizational Sciences. (2016)

Conference papers:

[1] Konecki, Mario; Tomičić, Igor; Milić, Luka: Natural language processing and artificial intelligence in everyday life. Proceedings of IAC-SSaH 2015: International Academic

Conference on Social Sciences and Humanities in Prague 2015. Prague, Czech Institute of Academic Education z.s., 2015. 239-243

[2] Oreški, Dijana; Konecki, Mario; Tomičić, Igor: Temporal Recommender System as a mean of helping customers in finding the right service. Proceedings of IAC-SSaH 2015. Kratochvílová, Helena, Kratochvíl, Radek (ur.). Prague, Czech Institute of Academic Education, 2015. 244-251

[3] Schatten, Markus; Tomičić, Igor; Okreša Đurić, Bogdan: Multi-agent Modeling Methods for Massively Multi-Player On-Line Role-Playing Games. Mipro 2015. Rijeka: IEEE, 2015. 1503-1508

[4] Tomičić, Igor; Ćorić, Ana: Information addiction. Proceedings of the 21st Central European Conference on Information and Intelligent Systems. Aurer, Boris; Bača, Miroslav; Schatten, Markus (ur.). Varaždin, Faculty of Organization and Informatics, 2010. 271-280