Software Ecosystems vs. Natural Ecosystems: Learning from the Ingenious Mind of Nature

Deepak Dhungana
Lero, University of Limerick
Limerick Ireland
deepak.dhungana@lero.ie

Iris Groher
Johannes Kepler University, Linz
Linz, Austria
iris.groher@jku.at

Elisabeth Schludermann
Vienna University of Technology
Vienna, Austria
elisabeth.schludermann@tuwien.ac.at

Stefan Biffl
Vienna University of Technology
Vienna, Austria
stefan.biffl@tuwien.ac.at

ABSTRACT
The use of the term *ecosystem* in the context of extensible software platforms and third-party developers or user communities has made us ponder about the similarities between software ecosystems and natural ecosystems. We therefore compare software ecosystems and natural ecosystems to present an agenda for further research by analyzing some key characteristics of both types of ecosystems. We discuss the regulatory factors and mechanisms existing in nature, and then deduce key challenges that need to be dealt with, in order to achieve healthy operation of software ecosystems.

Categories and Subject Descriptors
D.2.9 Management; D.2.11 Software Architectures - *Domain-specific architectures;* D.2.13 Reusable Software - *Reuse models*

General Terms
Management, Measurement, Design, Economics, Experimentation, Human Factors, Theory.

Keywords
Ecosystem, Software, Nature.

1. INTRODUCTION
Nature has developed numerous ingenious solutions in the course of the evolutionary process [1]. Scientists in engineering contexts and engineers often try to learn from their natural counterparts and solve problems by following approaches in nature. Software engineering (SE) has already exploited many phenomena in nature to improve the efficiency of algorithms, tools, models, and processes. For example, the theory of evolution serves as the role model for genetic algorithms, where natural selection is applied to computer programs and data [6]. Ant colony optimization has been inspired by ants and their behavior of finding shortest paths from their nest to sources of food [3]. Other examples are neural networks, swarm robotics, and bee algorithms.

Recent developments and trends in software product line engineering have made us ponder about the term *software ecosystems*. It is clear that the term was coined to reflect the organization of software vendors, third-party developers, suppliers, and users [4]. The name is obviously deduced from the archetype *natural ecosystem*. However, not much work has been reported, which compare characteristics of natural and software ecosystems. Several research reports have emphasized the importance of considering software ecosystems from a business perspective [7]. However, key questions are how the two kinds of ecosystems can be mapped to each other, and how software ecosystems researchers and practitioners can benefit from insights taken from natural ecosystems.

Successful initiation, management, and monitoring of ecosystems remain big challenges for software ecosystem practitioners. This is partly because the ecosystems community still lacks proper management theories, tool support, and consolidated experience in this area. As software ecosystems have emerged as a paradigm for maintaining large scale software product lines [4], many researchers and practitioners attempt to apply tools and techniques from software product line engineering (SPLE). However, we argue that the challenges related to software ecosystem management are more far-reaching than the scope of traditional SPLE [11]. Researchers and practitioners could benefit by understanding how nature deals with similar issues in complex natural ecosystems. For example, a balanced *natural ecosystem* is the result of delicate interplay between all participating components, e.g., interactions between the biota and their physical environment as well as interactions between organisms. Major functional aspects of the system such as (i) the amount of primary energy that is produced by photosynthesis, (ii) how energy or materials flow along the many steps in a food chain, or (iii) the rate at which nutrients are recycled in the system determine the health of an ecosystem. Our research goal is to derive lessons for software ecosystems, based on such observations in nature.

In this paper we introduce a simple model that characterizes ecosystems and maps aspects of ecosystems in software and nature. Based on discussions with a domain expert in biology and standard literature on natural ecosystems we focus on resource management (recycling or reuse) and optimization of processes within the ecosystem to increase value of the products or size of the

ECSA'10, August 23–26, 2010, Copenhagen, Denmark.

ecosystem population. We identify insights from natural ecosystems regarding resource management and sustainability that may be relevant for software ecosystem managers and discuss future research issues.

This paper is organized as follows: in Section 2 we present examples of natural and software ecosystems. Section 3 presents sustainability as one of the fundamental challenges of ecosystems. We then compare the two ecosystems in Section 4. In Section 5, we present a framework for sustainable ecosystems based on our observations. In Section 6 we provide a research agenda for software ecosystems. Section 7 concludes this paper with a summary and a description of further work.

2. EXAMPLES

The term ecosystem usually refers to the operation of the system as a whole [1]. However, each individual plays an important role in the overall stability and sustainability of an ecosystem. Here we present two examples, one from each kind of ecosystem to exemplify the role of different participants.

2.1 Natural Ecosystem

For illustrative purposes in this paper, we define the scope of natural ecosystems to be habitats, e.g., grassland with a stable animal and plant community. Our examples come from Savanna grasslands, which are found in the centre of continents between 5 and 15 degrees north and south of the equator. The vegetation consists of grasslands with some woodland with isolated trees such as Baobab and Acacia. Temperatures are high throughout the year averaging around 28 degree centigrade and there are clear seasons of wet and dry weather. Rainfall is convectional with very heavy downpours.

Important aspects : Vegetation must survive a long period of drought and the fires during the dry season. As a reaction, grass grows quickly - up to 130cm high during the short rainy season. Baobob trees store water in their trunks and have thick bark which protects against fire and reduces moisture loss. Trees have few leaves, so less moisture is lost through evaporation and transpiration - leaves are small, waxy, and thorny to reduce moisture loss. Trees lose their leaves through the dry season to conserve moisture. Trees have long roots so as to extract water from the deep ground [15].

2.2 Software Ecosystem

As an example of a software ecosystem we take the well-known Eclipse platform [10]. Eclipse is an open source community, whose projects are focused on building an open development platform comprised of extensible frameworks, tools, and runtimes for building, deploying, and managing software across the lifecycle. The project was created in 2001 by IBM and supported by numerous software vendors (e.g., Borland, Rational Software, Red Hat, SuSE, TogetherSoft). By the end of 2003, this initial consortium had grown to over 80 members. The Eclipse Foundation was created in 2004 as an independent non-profit organization acting as the steward of the Eclipse community [10].

Important aspects: Eclipse community and Eclipse Foundation are involved in active marketing and promotion of Eclipse projects and the wider Eclipse ecosystem. An ecosystem that extends beyond the Eclipse open source software (OSS) community to include commercial products based on Eclipse, other OSS projects using Eclipse, training and services providers, magazines and online portals, books, are all key to the success of the Eclipse community [10]. To assist in the development of the Eclipse ecosystem, the Eclipse Foundation organizes a number of activities, including marketing events with Member companies and community conferences. All technology and source code provided to and developed by the Eclipse community is made available royalty-free via the Eclipse Public License [10].

3. SUSTAINABILITY AS A CHALLENGE

In this paper, we particularly focus on the challenge of *nurturing ecosystem sustainability factors* . We have chosen this issue because it is fundamental to both kinds of ecosystems.

A *sustainable* natural *ecosystem* maintains its characteristic diversity of major functional groups, productivity, and rates of bio geochemical cycling, even in the face of disturbing events [5]. In nature, the sustainability is governed, among other factors, by biodiversity and the balance of geochemical resources [8].

Analogous to natural ecosystems, we **define** a *sustainable software ecosystem* to be the one that can increase or maintain its user/developer community over longer periods of time and can survive inherent changes such as new technologies or new products (e.g., from competitors) that can change the population (the community of users, developers etc) or significant attacks/sabotage of the ecosystem platform. The identification of factors that contribute towards fostering sustainability of software ecosystems is a big challenge, as it involves aspects from different disciplines (e.g., business, sociology, or law). The participants of software ecosystems (vendors, users, or communities) face the challenge of meeting growing demands for producing/using affordable, high quality software in short time [13]. Also, decision-makers in the companies or communities must balance economic growth and community development with conservation of code quality, system architecture, and minimal use of resources.

Depending on the kind of ecosystem, there may be different factors that contribute to long-term sustainability of the ecosystem. Initiation of an ecosystem initiative therefore needs to identify ecosystem sustainability factors and enforce these without imposing excessive control. Effective management of ecosystems will require actions at all scales, from the local to the global [8].

4. COMPARISON OF ECOSYSTEM TYPES

Software ecosystems are made of software vendors, suppliers, and users plus the socio-economic environment, including a regulatory framework [7]. The term *natural ecosystem* refers to the combined physical and biological components of an environment. An ecosystem is generally an area within the natural environment in which physical (abiotic) factors of the environment, such as light, water and soil, function together along with interdependent (biotic) organisms, such as plants and animals, within the same habitat to create a stable system.

Despite the apparent differences, one can find parallels between the two “worlds”, for example:

- Both ecosystems have a **reservoir or finite resources**, which requires subtle “housekeeping”. For example, in natural ecosystems, critical resources like energy (food), minerals (nutrients), or water are implicitly managed by the laws of nature. In software ecosystems critical resources are platform architecture, code, time, money, users, developers, etc. Regulatory frameworks are required for the proper management of these resources.

- The **population** of natural ecosystems is controlled by incentive for individuals to enter or leave natural ecosystems (e.g., better breeding places, availability of food etc). This can be compared to incentive for users to buy/use/extend software ecosystem products. Population control (to make sure, there is a balance between producers and consumers) is important in both ecosystems.
- **Interactions** between the participating units e.g., competition, symbiosis in nature can be compared to the collaboration and competition (between suppliers of same functionality) in software ecosystems.
- Transfer of **energy** between individuals in natural ecosystems (food chains, webs) **can be compared to** the transfer and translation of **knowledge** between different stakeholders in software ecosystems.
- **Processes:** Life cycles of natural ecosystems (population size, biodiversity) can be compared to market cycles (e.g., larger mobile phone sales in the beginning of academic calendars) and technological advances in software ecosystems.
- **Adaptation of individuals and the overall system to changes** of context factors in natural ecosystems can be compared to stakeholders' response to accept, reject, or "learn to like" software products in a software ecosystem.

We particularly focus on two aspects (resource management and biodiversity) in this paper.

4.1 Resource Management

A natural ecosystem balances its resources by recycling and reusing them whenever possible. Ecosystems would otherwise not sustain because resources such as water (with some exceptions like aquatic ecosystems), and nutrients **are scarce**. The participants of the ecosystem produce and transform energy along the different stages by using available resources. Recycling processes are triggered by microorganisms and occur at all energy levels to reincorporate nutrients in the ecosystems resource pool. This can be mapped to common settings in software ecosystems, where the stakeholders (software vendors, suppliers, developers, testers, users, etc) produce and/or use the ecosystem resources (code, COTS components, documentations, licenses, architecture descriptions, models, etc.). **The resources available in the ecosystem** are therefore crucial for the sustainability in both types of ecosystems. For example, the productio**n of energy through photosynth**esis is a precious task for all ecosystem participants, just like the development of extensible or adaptable software frameworks through careful design and implementation in software ecosystems.

One can observe the following aspects as foundation for a working ecosystem both in biology and software:

Technology. Biological and technical interfaces, e.g., as basis for developing/breeding the next generation; for finding/using energy sources.

Economic cost/benefit considerations, which balance scare resources with best benefits; for deciding on alternatives in foraging, breeding, and choosing living space.

Cycles in Natural Ecosystems

Energy is continually put into a natural ecosystem in the form of light energy [5]. Energy is moved through an ecosystem via a food web, which is made up of interlocking food chains. Energy is first captured by photosynthesis (primary production). This energy is transferred and translated into other forms, as it wanders through different food chains. Nutrients (chemical substances), are recycled within an ecosystem by biogeochemical cycles (e.g., nitrogen cycle, carbon cycle **and phosphorous cy**cle).

Cycles in Software Ecosystems

Just like in natural ecosystems, software ecosystems need a continuous input of energy in the form of new development or maintenance of the ecosystem. Vendors of the ecosystem platform invest huge amounts of resources (money, time, manpower) to initiate a software ecosystem. Just like the food chains and webs in natural ecosystems, one finds chains of knowledge in software ecosystem. The knowledge about the platform architecture wanders through the knowledge chain (it is transformed and translated as needed) to developers and users of the ecosystem products. In software ecosystems, technology is the basis for developing (breeding) the next generation of the software product or attracting a new community of users.

The Eclipse community and especially the committers of the various projects put energy into the Eclipse ecosystem in form of commits to the resource repository. Participating companies even invest money into the ecosystem. Figure 1 shows the number of commits to the repository by topic by year. Most topics have growing numbers of committers. Knowledge is transferred first by the Eclipse Foundation that organizes member activities and by the large amount of offered online training courses, articles, documentation, and portals.

	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010
birt					44036	52744	35412	23939	27818	5135
datatools	35	8	10	6	9719	15330	32315	18197	12306	2822
dsdp						11162	15239	19807	14974	7718
eclipse	183345	276084	266690	354073	377545	268323	191120	230369	199972	62122
modeling			82	31575	66614	106869	162033	215224	357110	206437
rt			1144	26054	73894	36846	184731	516865	248005	109943
soa							2225	1101	34900	1049
stp						67798	95938	52310	16077	9871
technology		3209	5586	13454	23567	81825	227860	437820	231494	41242
tools	7	13012	32772	72296	129540	87177	133238	141286	156816	84834
tptp		20	39739	54541	145582	110872	55351	36509	14294	4522
webtools			1510	55786	168142	112207	195083	216203	397771	36499

Figure 1. Eclipse Commits by Topic by Year [12]

4.2 The Role of Biodiversity

Biodiversity plays a crucial role in ecosystem resilience (the power of an ecosystem to bounce back from a disturbing event), when an ecosystem is disturbed by spreading risks, making it possible for the ecosystem to reorganize after disturbance. Ecosystems seem to be particularly resilient if there are many species performing the same essential function (such as photosynthesis/primary producers, decomposition, or population control/predators) and if species within such "functional groups" respond in different ways to disturbances. Then, species can compensate for each other in times of disturbance.

eclipse	IBM	69%	90%	84%	86%	84%	87%	78%	85%	91%	86%
	Innoopract								1%	1%	
	InstantiationsInc.										1%
	IntelCorporation								1%	1%	1%
	Oracle					1%	1%				
	ProsystSoftware							1%	1%		
	RedHatInc.						1%	1%	3%	1%	1%
	ResearchInMotion			1%	1%	1%	1%	1%	1%		
	SAPAG										1%
	SAS				1%	1%	1%				
	SiemensAG									1%	1%
	Sonatype			1%	1%	8%	1%	1%	1%	1%	1%
	Soyatec									1%	1%
	Tasktop										1%
	WeigleWilczekGmbH								1%		
	ZendTechnologies									1%	
	compeopleAG							1%	1%		
	individual	9%	5%	9%	4%	6%	9%	16%	8%	4%	2%

Figure 2. Eclipse Project Diversity (Percent of Lines of Change by Year) [12]

In software ecosystems, diversity is equally important. Diversity here means the involvement of different developer and user groups. For example, through support for different programming languages, platforms, and hardware devices, one creates a larger set of producers of software artifacts. In the same way, by supporting different user groups (e.g., different age groups, different domains, or different roles of the involved users), one can considerably increase the diversity in software ecosystems. The loss of a community can be better compensated when such diversity is inherent in software ecosystems. In addition to diversity, it is equally important to carefully plan and consider business resumption and continuity plans for collapse of the software ecosystem.

In the context of Eclipse Ecosystem, both companies and individuals from all over the world use Eclipse and take part in its development. The Eclipse ecosystem thus involves many different developer and user groups. Eclipse is based on Java but includes a wide range of plugins from many different topics. It is not only a successful software development platform but supports modeling, web and enterprise development tools, service-oriented architectures, and data management. Figure 2 shows the project diversity of the Eclipse core plugin by company by year (from 2001 to 2010).

5. LESSONS LEARNED

Our observation reveals that, on a high level, similarities between the two kinds of ecosystems seem obvious; however, not all characteristics of natural ecosystems can be mapped to software ecosystems. One of the key differences between the two is the fact that knowledge in software ecosystems is not a physical quantity like the nutrients or energy in natural ecosystems, i.e., the sender does not lose the initial knowledge, when she passes it to somebody else.

Nevertheless, there are some typical patterns of cycles, population growth or decline, adverse conditions and reactions in both ecosystems. Based on our discussion of the different similarities, we have come up with a management framework for software ecosystems to foster long term stability. As depicted in Figure 3, a sustainable software ecosystem needs to consider three management

Biodiversity	Natural Ecosystems	Software Ecosystems	Example: Eclipse Ecosystem
What is it?	Variety of all forms of life, from genes to species	Variety in the developer and user communities	Both different companies as well as individuals from all over the world
Why is it important?	To enable optimal usage of ecosystem resources	To increase the value of the product by extending the market horizon	Different projects both commercial and open-source on different topics
How does it help?	When one species is disturbed, the critical ecosystem activities (e.g., photosynthesis) are carried out by others.	When one market segment is obsolete because of competitive technological platforms, the software ecosystem can still foster in other areas.	Possible because of the wide range of topics and projects
How to ensure it?	Nature ensures sustainability by sophisticated balance of resources.	Support a wide range of programming languages, platforms or hardware devices and a broad user community across domains or user groups.	Wide range of users and committers from all over the world. Domain is not restricted to software development but is the main focus (but many different areas within the software development domain)
What happens otherwise?	Ecosystem resources are not balanced, which results in slow degradation of the whole system.	The competitive advantage over other ecosystems degrades, resulting in a diminishing/vanishing community.	Advantage is that it is open-source and widely known and used. Used both in industry and academia.

Table 1: The role of bio diversity in different ecosystems. Examples from the Eclipse Ecosystem.

perspectives: technical issues, business consideration, and community participation.

From observations of natural ecosystems, we have learnt that a framework for a healthy ecosystem requires continuous feedback from the ecosystem on all the three fronts. Proper management of resources requires "biodiversity" and feedback loops. A monitoring framework needs to consider different parameters such as whether the variability of the ecosystem products is adequate to attract new developers and users, and whether the ecosystem resources are available such that they can be used by different participants efficiently (e.g., availability, usability, etc).

Each individual participant of the ecosystem contributes to the sustainability of the ecosystem by adhering to the fundamental laws governing the ecosystem. In natural ecosystems these are basic instincts of the individuals or the laws of physics and chemistry. Similar governing body is required in software ecosystems that monitor the different sustainability factors and alert the participants, when the conditions are degrading. The framework presented in Figure 3, therefore distinguishes between local management and global management (monitors). The regulatory effect can only be achieved if the global strategies are implemented by each individual on a level of the individuals.

6. RESEARCH AGENDA

Based on our comparison of the two ecosystems, we now present a research agenda for implementing the framework depicted in Figure 3.

Identify and enhance knowledge cycles

Analogous to the food chains and biogeochemical cycles in natural ecosystems, a software ecosystem can profit from its knowledge cycles. This means that the transfer of knowledge between the different participants (e.g., vendor to supplier to developer to user) must be enhanced by proper definition of "knowledge carriers" and translators along the way. Identification of such knowledge cycles is a challenge in itself. Enhancement of knowledge cycles by involving the different participants is another major research issue for further software ecosystems research.

Foster diversity of contributors and users

A successful ecosystem needs to maintain a healthy community by involving a wide range of participants—this enables the ecosystem to thrive even when it loses parts of its population, as other remaining participants continue fulfilling the basic roles. For example, the Eclipse platform is successful because it does not define itself to be an IDE. It is often described as a platform for everything and nothing in particular, which includes a wide range of users per definition.

Define and monitor health indicators

A healthy community should be sustainable, livable, viable, equitable and prosperous. The goal is to create communities of people empowering each other to stay well. There is still much to do in the area of defining health indicators and monitoring them to check the health of running ecosystems.

Wahyudin et al. [14] propose a model on measuring the balance of developer and user communities and derive a measure for "project

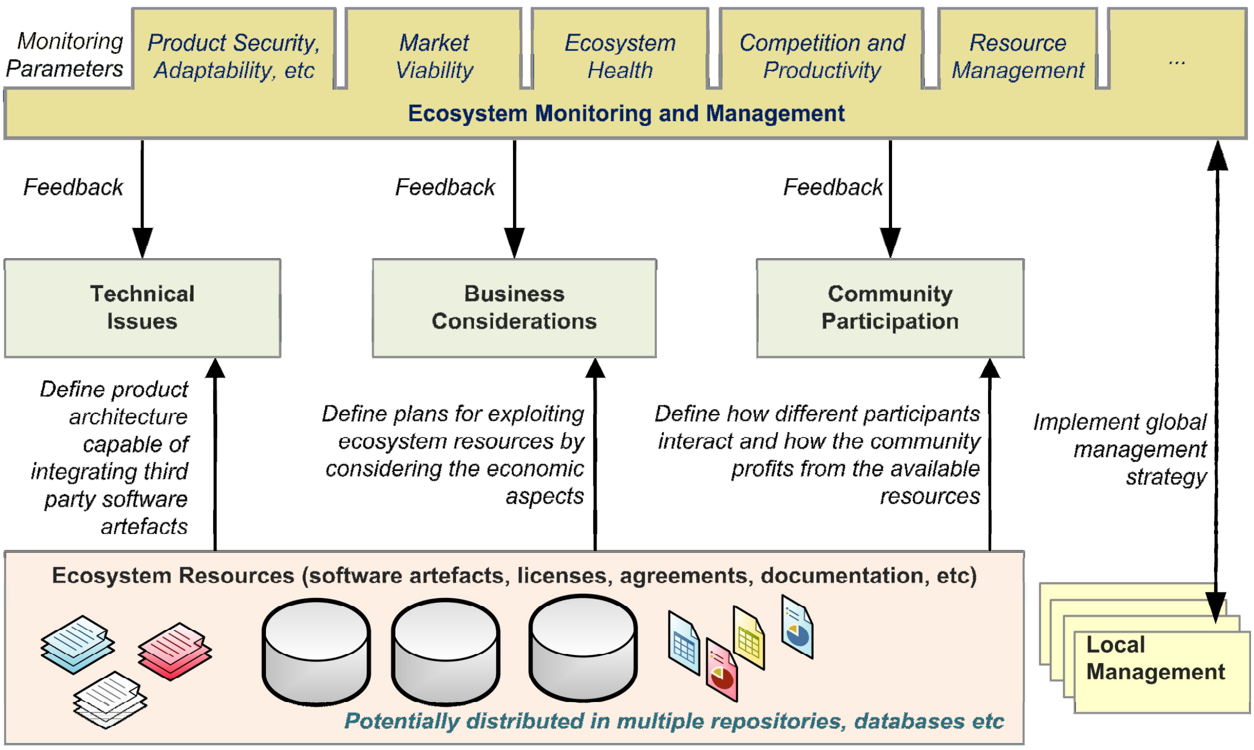

Figure 3. A framework for sustainable software ecosystem management.

health" of software projects. They validate the model with data from OSS projects in the Apache family by analyzing data from healthy and challenged projects in order to predict project health in the next few months of project life. In the scope of the study key indicators for project health were developer contribution (intensity of code commits, e-mails to common list, bug resolution) and "bug service delay", the time between a bug gets first reported and resolved.

Decision-making across organizational boundaries

Ecosystems may need to be "repaired" or new regulation may need to be applied after they are already functional. In such cases, a change requires a decision maker to enforce new rules across organizational borders. The decision making process itself needs to consider several aspects (technical, business, social etc) in order to not disturb the balance of ecosystems.

It may often not be possible to apply the same regulatory framework to all participating units of an ecosystem, as they operate in different legal contexts, which makes decision making across organizational borders even more difficult.

In natural ecosystems, different individuals or species in an ecosystem interact with each other through the resources they share. Individuals take decisions for themselves (local decision) and the overall effects are a sum of individual's decisions.

Infrastructure for fostering social interaction

When communities work together, a lot of communication between them occurs through the shared artifacts themselves, i.e., knowledge is attached to the artifacts and people communicate by sending them around. Such a scenario needs a more subtle infrastructure for social interaction, than the ones used today such as version control systems or shared workspaces.

As the different participants of an ecosystem may leave the community at any time, one has to make sure that the ecosystem does not suffer from knowledge drain problems. One way of solving such issues would be to lift the status of the shared artifacts to first class citizens. By building an infrastructure, where the artifacts "communicate" with each other rather than the people involved in the interaction, one could conserve knowledge in ecosystems.

In natural ecosystems, whenever plants or animals die, their bodies are reintegrated into the nutrient pool by micro organisms. This phenomenon makes sure that the ecosystem does not lose important resources and the energy budget stays balanced.

Adaptability of artifacts, processes, and stakeholders

In order to involve a large array of suppliers, developers or users in an ecosystem, ecosystems must provide a suitable environment for all the different groups. Variability of artifacts, processes, tools etc. is therefore an important issue to support diversity.

7. SUMMARY AND CONCLUSIONS

Many phenomena in software ecosystems can be in some way mapped to phenomena in nature. We have discussed some of them in this paper. The balance of energy through transformations is comparable to the creation and transfer of knowledge. To minimize loss of energy, one has to make sure that the knowledge produced in a software ecosystem can be efficiently consumed by other units to create another value. The balance of resources through biogeochemical cycles is comparable to resources in a software development environment, time, money, etc. These cannot be recycled but shared among the participants.

A species will move into an environment if the conditions become suitable for its establishment, and will move out of that environment if the conditions become unsuitable for its reproduction. Natural catastrophes like volcano eruptions, floods, or fires can disturb ecosystems, however most ecosystems have the power to regenerate and survive small disasters. Unnatural human interference, e.g., by using fertilizers, pesticides or, irrigation can imbalance an ecosystem or cause it to degenerate. Initiating and maintaining a productive software ecosystem is therefore about creating favorable environments for the developers and users. With the increase of the market value of the products being developed by the community as a whole, the ecosystem grows, thereby increasing the biodiversity of the ecosystem. This on the other hand increases sustainability of the ecosystem.

In the future, we plan to report on more detailed description of the study presented in this paper. Other aspects of comparison will include (i) the study of social hierarchies mapped to the hierarchy of software vendors, suppliers and user and (ii) territoriality in natural ecosystems mapped to privacy and intellectual property issues in software ecosystems.

It is important to note that the balance of natural ecosystems is such a complex maneuver, that it is impossible to directly apply all those concepts in the software world. However, it makes perfectly sense to understand the basic ideas and use them as guidelines, considering the (currently) early stage of research in the area of software ecosystems.

8. REFERENCES

[1] M. Begon, J.L. Harper, and C.R. Townsend. Ecology – Individuals, Populations and Communities, 2nd ed., Blackwell Scientific Publications, 1990.

[2] S. Biffl, A. Aurum, B. Boehm, H. Erdogmus, and P. Grünbacher (eds.). Value-Based Software Engineering. Springer, 2006.

[3] G. Bilchev and I. C. Parmee. The ant colony metaphor for searching continuous design spaces. In Selected Papers from AISB Workshop on Evolutionary Computing, pages 25–39, London, UK, 1995. Springer-Verlag.

[4] J. Bosch. From Software Product Lines to Software Ecosystems. Proc. 13th International Software Product Line Conf. (SPLC 2009), SEI.

[5] F. S. Chapin, M. S. Torn, and M. Tateno. Principles of ecosystem sustainability. American Naturalist, 148:1016–1037, 1996.

[6] D. E. Goldberg. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[7] S. Jansen, S. Brinkkemper, and A Finkelstein. Business Network Management as a Survival Strategy: A Tale of Two Software Ecosystems. 2009.

[8] J. Kay, H. Regier, M. Boyle and G. Francis. An Ecosystem Approach for Sustainability: Addressing the Challenge of Complexity, Futures Vol 31, #7, Sept. 1999, pp.721-742.

[9] D.G. Messerschmitt and C. A. Szyperski. Software Ecosystems – Understanding an Indispensable Technology and Industry. Massachusetts Institute of Technology, 2003.

[10] Eclipse platform, http://www.eclipse.org, 2010

[11] K. Pohl, G. Böckle, F. van der Linden, „Software Product Line Engineering. Foundations, Principles, and Techniques. Springer, Berlin, 2005.

[12] Eclipse project dash, http://dash.eclipse.org, 2010

[13] I. Sommerville, "Construction by Configuration: Challenges for Software Engineering Research and Practice," aswec, pp.3-12, 19th Australian Conference on Software Engineering (aswec 2008), 2008

[14] D. Wahyudin, K. Mustofa, A. Schatten, A. Tjoa, S. Biffl (2007): "Monitoring "Health" Status of Open Source Web Engineering Projects"; *International Journal of Web Information Systems*, 1/2, 3; S. 116 – 139

[15] W. H. Winter, "Australia's northern savannas: a time for change in management philosophy". in Patricia A. Werner. Savanna Ecology and Management: Australian Perspectives and Intercontinental Comparisons. Oxford: Blackwell Publishing. pp. 181–186., 1991.