Building an IT-ecosystem of services and exploiting agile methods

1st Tuomas Orama, 2nd Mika Lavikainen, 3rd Jaakko Rannila

1st, Development manager, Helsinki Metropolia University of Applied Sciences, Bulevardi 31 PL 4000, tuomas.orama(at)metropolia.fi
2nd, Project manager, Helsinki Metropolia University of Applied Sciences, Bulevardi 31 PL 4000, mika.lavikainen(at)metropolia.fi
3rd, Project manager, Helsinki Metropolia University of Applied Sciences, Bulevardi 31 PL 4000, jaakko.rannila(at)metropolia.fi

Keywords
Agile, Open source, SOA, Student Administration System, Ecosystem, Peppi, Software development

1. ABSTRACT

Metropolia University of Applied Sciences started to renovate Enterprise Architecture on 2010. The renovation process ended up to be a complete ecosystem for Higher Educational Institutions Today we are extending the ecosystem with additional services for students and student administration. To do this, we have started two projects that are implemented by using agile development methods. This paper explains the background of our ecosystem as well as the challenges in building and extending the ecosystem. One of the challenges that we have encountered is combining traditional project methods with agile methods and also doing this simultaneously in two separate projects. We also describe the methods that we have used to build our ecosystem as well as tools to manage and visualize the overall progress of the extension in our case projects.

2. INTRODUCTION

Traditionally, all the IT-systems used in Higher Educational Institutes (HEI's) have been specializing on one small part of business process. The process itself can be very extensive in the means of time or different process phases. For example, student management from applying to graduating could be described as one business process and still there might be several systems that must be used to handle this one process. From the user point of view every system delivers different user experience and, in the worst yet very common case, the user has to insert the same data several times so that these systems get all the necessary information. In the system administrational side there is a lot of headache in maintaining all the user permissions and data integrity.

The reason why it has come to that is mainly historical. IT-systems have been developing rapidly over time and different vendors have seized their territory in fulfilling the needs of a certain business process phase. In addition, technologies being used have been varying between vendors and there has been very little specified application interfaces to be used by others. It's still not the vendors or the IT-development to be blamed. The HEI's have not demanded or specified such things as Service Oriented Architecture (SOA), open interfaces or interoperability. Furthermore, business processes differ between organizations so it has been easier to build a system to support one small part of a process than to develop a system that could fit for the whole process on several organizations.

Also the methodology in developing IT-systems has been changing over time. Even a few years ago it

was almost a rule that IT-projects were implemented with traditional waterfall method where everything is strictly specified beforehand. The problem is that not everything can or should be specified in advance. Nowadays agile project management has become more popular in implementing IT-projects but it has also some drawbacks as explained later on.

In short, we have noticed some problematic issues in current IT-environment that we wish to fix in our new ecosystem. This paper explains how we are expanding our current ecosystem and how we are doing it.

3. ECOSYSTEM BACKGROUND

Renewing all systems in one project can be risky. That one project will be a mammoth. Rather than solving all problems in one project it could be wise to split that mammoth project to smaller pieces. That is exactly what we have done.

Six years ago we realized that it is not possible to go on like before. Many things had to change. We set up a long-term objective to renew all main systems for the end users.

First we had to change the way of thinking and forgetting the old habits. That process included

- Enterprise architectural thinking
- Creating and acquiring IT systems
- Using new technology and de facto standards
- Providing services based on user roles
- Committing executives

We made big decisions and commitments for future. Some of the decisions would carry on for many years to come on and we had to commit our executives for those decisions. Without executive commitment long-term projects simply wouldn't have a chance.

We started the change over 6 years ago and we are still half a away from The Goal. But saying so it is also important to say that we have already achieved many phasial goals along the way. We have already implemented the ERP services and they have been in production over 2 years now. Also we are in the verge of implementing student administration system and services for students to production.

3.1 Enterprise architectural thinking

Enterprise architecture (EA) thinking is getting foothold in organisations. EA is nowadays guiding many future IT projects and giving them objectives for the future. Enterprise architecture is defined by Gartner as follows "***Enterprise architecture (EA)*** *is a discipline for proactively and holistically leading enterprise responses to disruptive forces by identifying and analyzing the execution of change toward desired business vision and outcomes. EA delivers value by presenting business and IT leaders with signature-ready recommendations for adjusting policies and projects to achieve target business outcomes that capitalize on relevant business disruptions. EA is used to steer decision making toward the evolution of the future state architecture.*" (Gartner, 2015) As Gartner also defines, EA itself doesn't create future software but instead it gives value and objectives for projects that creates the future IT systems. This was something that we had to understand in the beginning. It is important to have EA but probably even more important to have projects that creates concrete values guided by EA.

3.2 Creating and acquiring IT systems

Normal or most common procedure of aquiring IT systems is defined in Systems development life cycle (SDLC) (Radack 2009). The most common phases of acquirements are:

1. Initiation - The need for system including system requirements
2. Acquirement/development
3. Implementation
4. Maintenance
5. Disposal

The most popular way in acquirement/development phase is to use Waterfall method (Bowes, 2014). SDLC and waterfall method were also very typical choices for us in 2009 and before that. In 2009 we also had to rethink the way of creating and acquiring IT systems. There were some weak spots on SDLC and waterfall. (CMS 2015)

The main character for both is that they assume that everything is known beforehand and nothing changes during the project. For example: 1. customers know everything that they need from the system before development phase and 2. all the technical choices made beforehand are the best ones for this project.

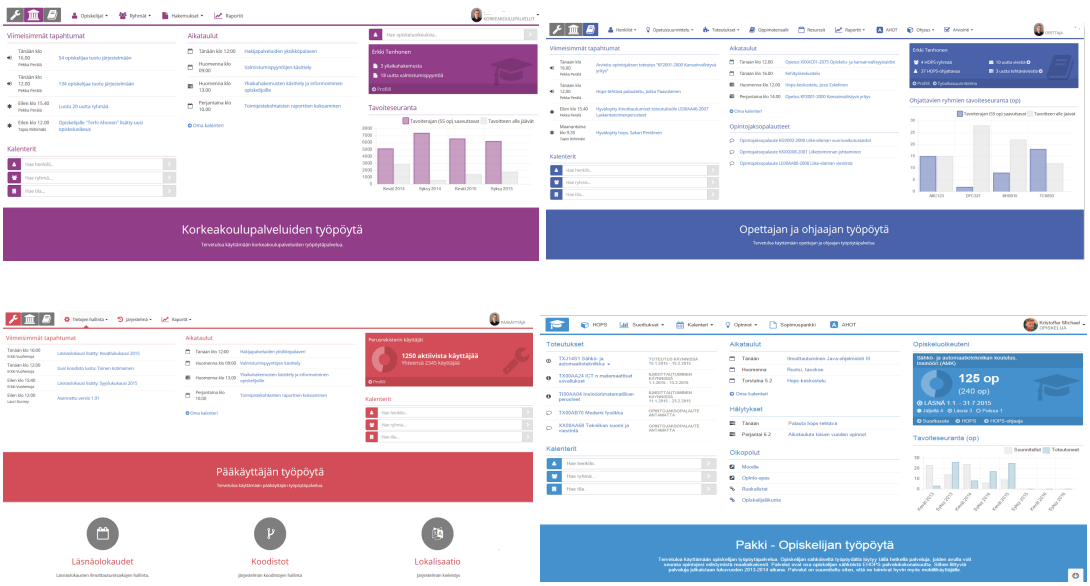
We had to think new ways that could tackle those weak spots. Choice was that we use agile approach for development phase and consortium for acquirement process. Agile methods helped us to make changes during the development phase and consortium gave us more tools for SDLC.

Rather than buying “blackbox” software from the market we wanted to create ecosystem of services that are defined, designed, developed and owned by us. This meant that we could also define the level of dependency to those technologies we chose to use. These choices gave us great advantage concerning the lifecycle phase of maintaining and further development. Referring news article on ComputerWeekly.com support and maintenance costs have increased every year for Global 2000 companies in the US and UK (ComputerWeekly.com, 2014). At the same time many CIOs expect that they will have IT budget cuts in the future (Gartner, 2014).

3.3 Providing services based on user roles

We knew at an early stage that the collection of services we were developing would cover wide range of business processes and some of the services would overlap between processes and user-roles. We decided to bundle needed services from the user-role point of view and build role based dashboards. Of course users may have multiple roles but that is not a problem since all the dashboards are on the same platform and changing a role on the fly is only one click away. There is no need to change between different systems. Also adding of the roles or making even new dashboards, is relatively simple, if necessary.

Basically all the users are using exactly the same services from the technical side. They only need to have some roles that they have access to the services. The user inherits some predefined permissions based on his/her roles and they are not bound to a specific dashboard. The permissions can also be fine tuned user-specifically. Almost all the business logic is built into the service layer which means that we can build role-optimized user interfaces very easily with the same background service. The role based dashboards look very similar (Picture 1) to each other although the actual services that they provide are meant to fulfill very different processes. Some services even appear exactly the same in different dashboards it they are needed by several roles - room reservation service is an example of this kind of service.



Picture 1: Role-based dashboards

Above are some screenshots of different dashboards. The idea is that every dashboard has same look and feel. Navigation and user experience are uniform through the ecosystem and more services can be added easily on the dashboard. Some services are fitted into the front-page like in the example, some services need more space and can be accessed from the top menu. The menu itself changes based on user role and user permissions.

3.4 Committing executives

Building an IT-ecosystem is a long-term decision both financially and operatively. This means that the organization has to be committed to the decisions. One point of view is financial, it may take several years to build an ecosystem piece by piece and also the already existing services will need some upgrades from time to time. In addition it is not usually feasible to adopt every new IT-solution that comes into the market especially if it does not fit into the chosen ecosystem. The executives need to think carefully when it is necessary to make exceptions on the ecosystem.

The other point of view is operational commitment. The organization needs to be committed to give necessary human resources in order to successfully carry out the whole process from setting the requirements to actual implementation. Also new procedures need to be introduced and put into use in the organization and this takes commitment.

4. EXTENDING THE ECOSYSTEM

What we call an ecosystem in this case includes the combination of architecture, platform and services provided to the end-user through role based dashboards. In short, the dashboard is actually just a collection of modular services that a certain user role needs to accomplish his/her tasks. The user can have several roles simultaneously thus having access to multiple dashboards. It is also possible to finetune user access within services themselves.

Student administration services is a project where we completely renew our old student- and accomplishment register including its user interfaces and reporting. The old system is outdated and incompatible with our ecosystem. Furthermore, we need wide range of application interfaces to the register in order to automate and develop new student services. This we cannot provide with our current, closed system.

Student dashboard services is a project that exploits the data and application interfaces created in the student administration services project. The project produce new services for student-roles including for example personal study planning and messaging services. It also combines existing services into the same role based dashboard.

As mentioned above, we have started two more projects in order to cover the missing elements: student administration services and student dashboard services. Student dashboard services -project is based on the requirements that have been defined earlier in nation-wide TIPTOP -project (TIPTOP, 2014). Student administration services on the other hand are being built on processes that were defined during the project. The requirements for both of the projects were made on 2013 and early 2014.

The actual development phase started on summer 2014. Both projects have their own budget, aims and steering group so they are completely separate projects although they implement the same ecosystem. It was mainly coincidental that the projects started almost the same time since the early background of both of the projects were very different and complex.

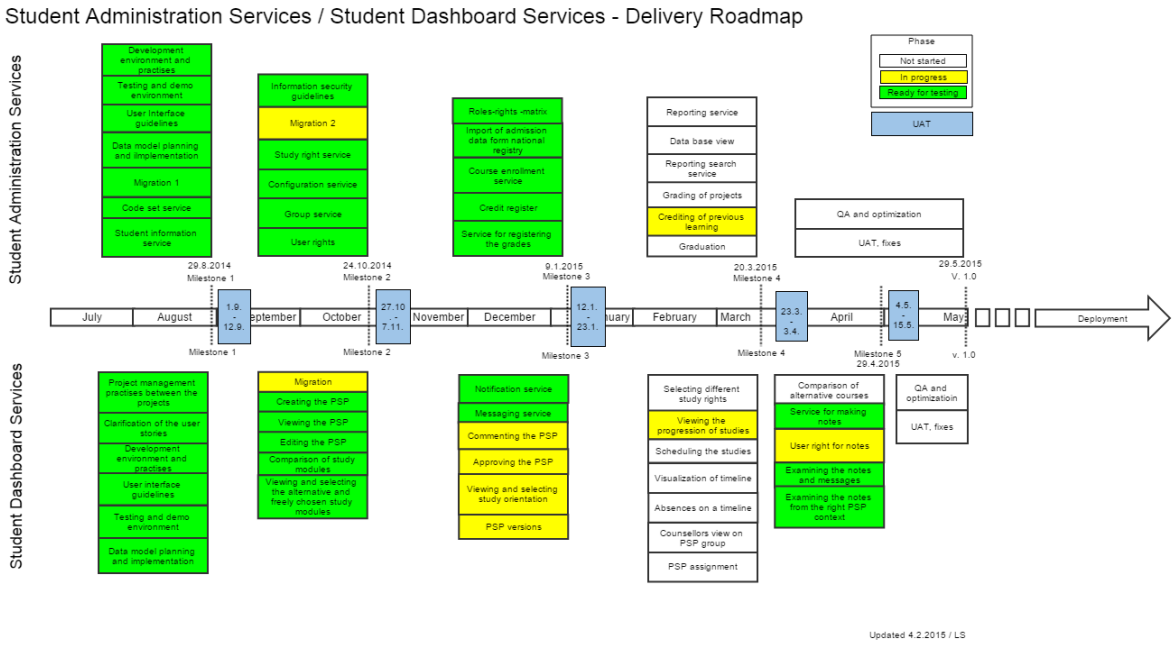
We were in a situation where we had relatively well defined ecosystem with a bunch of already implemented services and now we needed to extend the ecosystem in two separate projects. Shortly after starting the projects we noted that some project outcomes were dependent on each other and even some of the staff were working on both projects. We needed to somehow combine or synchronize these projects.

5. PROJECT METHODS AND OUTCOMES

On the bidding phase we ended up choosing same vendor for both projects. From the day one it became clear that the projects would be easier to manage if we would use the same methods in project management and development. From the vendors point of view it was rather challenging in the very beginning to hammer both projects to the same project model since the requirements were made with completely different way: the student registry services requirements were formulated on a very detailed way with a waterfall style, while the student dashboard services covered basically the user stories.

The vendor chose to use the scrum method in the development phase in both projects. Projects' timeline is divided in four or five milestones (producing a development version submitted to user acceptance testing) all having the own goals that fill the certain amount of the requirements. The actual development has been managed in two-week sprints including the established scrum elements, i.e. the use of product and sprint backlogs, sprint planning sessions, Daily Scrums, sprint review sessions and sprint retrospectives (SCRUM ALLIANCE, 2014). The chosen method has been suitable for our needs thanks to its consistency and transparency. It has also allowed us to follow the development and, on the other hand, clarify the requirements if - and when - needed.

The vendor has been using Atlassian JIRA's Agile features in the project management, and it has fulfilled our needs in managing the project and following the development. However, especially for the stakeholder needs, we have learned that one simple yet evolving picture is an excellent way to stay on the map of the project, and to communicate to stakeholders through the project's lifespan (Picture 2).



Picture 2: Projects' Delivery Roadmap

In addition to described scrum conventions, there have been a traditional project organization in both projects including the separate steering groups, joint project group, and a number of expert groups. Especially the expert groups have been crucially important in clarifying the requirements and testing the versions. Test results have been analyzed thoroughly and are used in shaping up the requirements or polishing the user interfaces and so on.

In the Student administration services project we started from the student administration services (information of students, study rights, accomplishment registry etc.) and administrator services (code set services, logs, localisation etc.), and then proceeded to teacher (user interfaces for approving the enrollments for courses and register the credits) and student services (enrollment for courses).

Student dashboard services project, on the other hand, has been producing features for personal study planning (PSP) for both the students' and study counsellors' dashboards. The PSP tool covers features for both "narrow PSP" and "open-ended PSP" needs. (Ansela & al., 2006). The narrow PSP refers to the target of making a concrete plan 'what to study', 'when to study' and 'how to study'. The open-ended PSP, on the other hand, is more focused on the learning and interacting with the student counsellors and peers. That is why the messaging service has been essentially integrated to the personal study planning service.

As explained above, these two project are producing a certain amount of new services in four different user role dashboards: student, teacher/counsellor, student administration, and administrator. This highlights the benefits of service orientated architecture. We can gradually build a compatible ecosystem using the existing application interfaces and services.

5.1 Traditional versus Agile methods

Agile development methods have been lately very popular in IT-projects. Yet, there has been a lot of debate on how effective those agile methods really are. In our case we are using agile SCRUM for the actual coding part of the projects and more traditional project management for the overall implementation. By overall implementation we mean the specification phase, coding/development, training and actual deployment phase. There have been numerous empirical researches on agile development and the benefits or drawbacks it might bring compared to non-agile development. The results do not seem to be conclusive and it is not always clear to the developers themselves if agile methods are being used or not (Murphy, 2013).

On the basis of our experience the question should not be if we are using agile methods. More important is that we use the right methods. We have found a way to use agile methods with our vendor for the development part of the project and combine it to a more traditional project model which fits better for the more rigidly governed Educational Institute.

5.2 Future challenges of the ecosystem

We have described how we have built and started to extend our IT-ecosystem of services. Yet, we acknowledge that the chosen ecosystem will not last forever and needs to be replaced at some point. Currently used architecture supports renewing of modules and services quite freely but still it has some limitations. There will always be new technologies to replace older and we do not know the actual lifespan of our architecture model. Our ecosystem covers major part of our business processes so it has become crucial part of our organization. When the day comes that we need to replace it we need to think it through carefully. Still, we think that we will be in a better situation compared to building current ecosystem since now we have modules and services that are well documented and implemented with identical technologies. Since we have built our current ecosystem modularly over time we should be able to renew it with new technologies accordingly.

6. REFERENCES

Ansela, Haapaniemi, Pirttimäki, 2006: Personal Study Plans for University Students. Retrieved on February 12, 2015. https://www.uef.fi/documents/1526314/1526339/Personal+Study+Plans+for+University+Students+-+A+Guide+for+Study+Counsellors_Ansela,%20Haapaniemi,%20Pirttim%C3%A4ki_2006.pdf/e7a08615-db67-42c1-b9ae-14397153a72f

Bowes 2014, *Agile vs Waterfall: Comparing project management methods*. Retrieved on February 11, 2015. from: http://manifesto.co.uk/agile-vs-waterfall-comparing-project-management-methodologies/

Centers for Medicare & Medicaid services (CMS) 2005, Selecting a development approach. Retrieved on February 11, 2015. from: https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/SelectingDevelopmentApproach.pdf

ComputerWeekly.com website, 2014, *Application support and maintenance costs increase 29%*. Retrieved on 11, 2015. from: http://www.computerweekly.com/news/2240216589/Application-Support-and-Maintenance-ticket-costs-increase-29

Gartner website, 2014. *Gartner Says More Than a Quarter of Federal and National Government CIOs Anticipate Budget Decreases in 2014*. Retrieved on February 11, 2015 from: http://www.gartner.com/newsroom/id/2731317

Gartner website, 2015. *IT Glossary - Enterprise Architecture (EA)*. Retrieved on February 9, 2015. from: http://www.gartner.com/it-glossary/enterprise-architecture-ea/

Murphy, 2013, Have Agile Techniques been the Silver Bullet for Software Development at Microsoft?. Empirical Software Engineering and Measurement, 2013 ACM / IEEE International Symposium on [0-7695-5056-8], pg:75 -84

Radack, 2009, *The system development life cycle (SDLC)*. Retrieved on February 11, 2015. from: http://csrc.nist.gov/publications/nistbul/april2009_system-development-life-cycle.pdf

Scrum Alliance website, 2014, Retrieved on February 12, 2015. from: https://www.scrumalliance.org/why-scrum/core-scrum-values-roles

TIPTOP project website, 2014, Retrieved on February 11, 2015. from: http://web.csc.fi/sivut/tiptop

7. AUTHORS' BIOGRAPHIES

Tuomas Orama works as development manager and is the head of the development unit in Metropolia University of Applied Sciences in Helsinki, Finland. He graduated as an industrial designer from Kuopio Academy of Design. He has studied also in Institute of Design and Fine Arts in Lahti university of applied sciences and in università per stranieri di Perugia. His work experience includes dozens of IT-projects for more than a decade. He has worked in several expert positions in national IT-projects in HE-level.

Mika Lavikainen works as project manager at Metropolia University of Applied Sciences in Helsinki, Finland. He has master's degree in science (Industrial Engineering and Management, Lappeenranta University of Technology, 2005). His work experience includes several IT-projects varying from CRM-projects to fully tailored software as a service projects as well as large EU Framework six projects. In addition to IT-based projects he has experience in developing advanced collaborative working environments (including augmented reality prototyping), Collaborative Networked Organizations (CNO's) and virtual organizations.

Jaakko Rannila works as project manager at Metropolia University of Applied Sciences in Helsinki, Finland. He has bachelor's degree in science (Industrial Engineering and Management, Helsinki Stadia polytechnic, 2006). His work experience includes several IT-projects varying from SOA-projects to fully tailored software projects as well as large national project involving Education IT-management. In addition he has experience in developing ERP systems and search engine systems in Higher Education, renewing qualification criteria to upper secondary education in degree programme of Information Technology in a Finnish National board of Education project.