# Scrum for Product Innovation: A Longitudinal Embedded Case Study

J. M. Bass[1], A. O. Abdul[1], H. Ghavimi[2], N. MacRae[2], P. Adam[2]
[1]*University of Salford*
[2]*Add Energy Ltd*

## Abstract

*This article describes the tailored product innovation processes used in a partnership between Add Latent Ltd., an asset integrity and maintenance management consulting services provider in the energy sector and University of Salford. The challenge faced by the company is to make their in-house expertise more readily available to a worldwide audience. A longitudinal embedded case study has been used to investigate how installable desktop software applications have been redesigned to create a new set of cloud hosted software services.*

*The innovation team adapted an agile scrum process to include exploratory prototyping and manage the geographical distribution of the team members. A minimum viable product was developed that integrated functional elements of previous software tools into an end-to-end data collection, analysis and visualisation product called AimHi which uses a cloud-hosted web services approach. Field trials were conducted using the software at the Uniper, Isle of Grain power station in Kent, UK. Enhancements were made to the AimHi product which was adopted for use at the Uniper site. The product emerged from a Knowledge Transfer Partnership which was evaluated on completion by Innovate UK and awarded the highest possible "outstanding" grade.*

*Extended periods of evaluation and reflection, prototyping and requirement refinement were combined with periods of incremental feature development using sprints. The AimHi product emerged from a technology transfer and innovation project that has successfully reconciled conflicting demands from customers, universities, partner companies and project staff members.*

## 1. Introduction

This article describes the innovation process used to create a new product, implemented as a cloud-hosted software service, over a 5 year period. The product calculates key performance indicators for asset maintenance engineers in the energy sector.

In 2005, 15 plant workers were killed when a hydrocarbon vapour cloud exploded at the BP America Texas City refinery. While the immediate cause of the disaster was ignition of the vapour cloud, probably by a vehicle engine, the contributing factors leading to the vapour escape was under-investment in the refinery infrastructure and deficiencies in the asset integrity programme [1].

To avoid such catastrophes, Add Latent provide asset integrity consulting services to client companies. To support these consulting services several software tools have been developed for internal use. This article describes the evolution of these internal products into cloud-hosted software services with the aim of attracting a wider customer-base. A field trial of the software was conducted at a UK combined-cycle gas turbine power station producing electricity for about 1 million homes. Following the field trial enhancements were made and the software adopted for use by the power station operator.

The article contributes new knowledge regarding the innovation process informed by lean and agile software development methods used. The rest of the article is structured as follows. In the next section related work is discussed. In Section 3 the research methods adopted in the study are presented. The case study describing the evolution of the AimHi product is presented in Section 4. Section 5 discusses the case study findings, while conclusions are presented in Section 6.

## 2. Related Work

### 2.1. Innovation and Technology Transfer

There are a number of motives for companies and universities to work together to enhance innovation. For universities there are opportunities to gain exposure to practical problems enhancing business relevance, test the application of ideas in industry, access funding for research and promote innovation. For companies, there are opportunities to enhance technological capacity, enhance competitiveness as well as to access new knowledge and complimentary know how [2].

National and regional innovation systems have evolved to include corporate, digital, district- and university-based innovation systems [3]. Corporate innovation systems have emerged using open innovation processes. Digital innovation systems have built-up around mobile `app.' stores. While municipalities have developed innovation support strategies usually targeting smaller companies.

## 2.2. Cloud-hosted Software Services

Add Energy identified five main benefits of cloud hosting software services:

- Support for fluctuating demand profiles through scalability,
- Centralized live code base to support software maintenance and evolution (compared with application deployment on client workstations),
- Pay-as-you-go payment model based on actual service usage compared to up-front capital investment for servers anticipating expected demand,
- Access to robust redundant infrastructure beyond the investment resources of most small business, and
- Single point to manage security updates and patches to mitigate the risks of `malware' and malicious attacks.

These benefits can be combined with virtualised deployment to encapsulate functional code and all dependencies into a single container [4]. Live code environments can be more easily migrated from one server to another (for example, if hardware upgrades are required) when compared with native deployment on a base server. Further, clients with highly sensitive data may choose to create private cloud deployments behind their own firewalls rather than use a public cloud hosting deployment.

## 2.3. Agile Development Methods – Scrum

Agile methods emerged in the 1990s to reduce risk of project failure and increase flexibility in the face of changing requirements [5].

There are a range of software development methods, considered as being agile, including Dynamic Systems Development Methods [6], Crystal [7], Extreme Programing (XP) [8] and Lean Software Development [9].

These techniques share the concept of short iterations that provide regular and frequent feedback. For development teams, these short iterations help identify the causes of delays and bottlenecks in the development process. For external stakeholders' short iterations provide visibility of progress and regular opportunities to influence the direction of the team, reflected in changing priorities.

XP provides a series of engineering practices including test-driven development [10], refactoring [11], continuous integration [12] and pair programming [13]. XP encourages customer involvement through an on-site customer. Development is performed in a series of short iterations using incremental design and incremental planning. XP encourages a focus on developing working software without the need for unnecessary reports and documentation.

In recent years, however, a trend has been observed towards practitioner adoption of scrum [14]. Scrum has been advocated as a method for small software development teams [15]. Scrum inherits a number of practices from XP but focuses on the orchestration of agile project teams: sprint planning at the start of an iteration, daily coordination meetings, sprint reviews and retrospectives at the end of an iteration. Many of these agile practices have been studied extensively [16].

These agile development processes are described from three perspectives: roles, ceremonies and artefacts.

**2.3.1. Scrum Roles.** Conventionally scrum comprises three roles: Product Owner, Scrum Master and the self-organising Feature Development Team [17]. The product owner is responsible for identifying and prioritising requirements for the system under development. In large-scale project product owners work to support cooperating development teams [18]. Product owners work with clients, or perform market research, in order to elicit new requirements. The requirements collected are often in the form of user stories. The user stories are collected into a prioritized list, known as a product backlog. High priority items are taken from the product backlog to create a smaller, so called sprint backlog, for each development increment. Product owners also define the acceptance criteria for each requirement and decide if an implemented and tested feature meets the acceptance criteria, and hence is ready for release.

Scrum masters act as agile method mentors for teams and work to remove any impediments that obstruct development team progress [19]. Scrum masters facilitate the development process and not undertake team leadership or project management activities [20]. Scrum masters facilitate ceremonies such as: daily stand-up coordination, sprint planning, retrospective and customer demonstration meetings; they also facilitate integration of new feature code in to the common code base trunk. Scrum masters also help those outside the scrum team find the best ways to communicate with team members.

The self-organising team is organised as a feature team [21]. Feature teams are responsible for the software architecture, design, implementation and testing of each feature. Team members work together to produce estimates of effort required to produce features defined by user stories in the backlog. Scrum teams are small, typically with no more than nine team members, in order to be nimble yet large enough to accomplish meaningful tasks.

 

**2.3.2. Scrum Ceremonies.** Scrum teams work in iterations that last between two- and four-weeks duration. Increments in scrum conventionally start with a planning meeting in order to produce estimates of the highest priority items in the backlog. These estimates produced by the team determine how many user story requirements can be accommodated within the sprint.

All team members participate in daily coordination meetings [22] answering three questions: `what have I done since yesterday's meeting?' `what will I do today?' and `am I facing any impediments?' Some teams introduce a fourth question: `will I create any impediments for others in the team?' Impediments are obstacles, often a lack of information, preventing team members from making progress with a task. The stand-up meetings are conventional conducted at a whiteboard. The whiteboard displays the status of the project with tasks shown at their various stages of development. At the stand-up meeting is conducted, scrum masters move work items that are completed onto the the next stage on the whiteboard.

At the end of each increment is a sprint review and retrospective meeting. The sprint review comprises a demonstration of newly implemented functionality to the customer (usually the product owner). The demonstration allows the product owner to decide if features are ready to ship into a live release of the product. After the demonstration, the team members review the increment. Usually each team member produces three positives and three negatives about the sprint. These are collated, and any common themes are used to create a set of action points for the team to improve upon for the next sprint.

**2.3.2. Scrum Artefacts.** Agile methods aim to focus on the production of working code [23]. That means aiming for a reduced set of development artefacts such as reports and documentation. However, this should not be interpreted as meaning that no documentation at all is produced, since design decisions and architectures (for example) must be recorded and disseminated to project stakeholders [24].

The artefacts produced by scrum teams routinely include: scripts, configuration files, source code, backlogs and burndown charts. Backlogs are lists of items representing requirements; conventionally prioritized by the product owner. Burndown charts provide a visual representation of the progress team members are making during each sprint.

## 3. Methods

This research is exploratory, comprising an embedded case study over a 5 year period [25]. In order to understand innovation using software application-level cloud architectures and agile development processes, we conducted a case study in Add Latent Ltd with the company research and development team as the embedded unit of analysis. Unlike hypothesis-testing, where the detailed distribution of variables is explored statistically or experimentally, this study relies on theoretical sampling [26]. This type of industrial case study is in the collaborative research tradition [27].

Case studies are suitable for exploratory research in which a hypothesis describing some phenomena is developed [28]. The longitudinal embedded case study approach was employed in order to provide a holistic, in-depth, analysis of one setting and are characterized by production of rich and detailed descriptions [29].

### 3.1. Research Sites

The company was selected from a population of engineering technology small and medium sized companies providing knowledge-intensive business services [30] using convenience sampling [31].

### 3.2. Data Collection

The research draws on records of over 100 project team meetings. The team meetings are usually conducted weekly during active periods and are minuted for internal project reporting purposes. During the Knowledge Transfer Partnership phase, a quarterly Local Management Committee meeting is held and professionally minuted in the presence of a representative of the funding agency. Workshops involving the project team and other stakeholders such as consulting engineers from the asset integrity domain and product marketing specialists have also been conducted and minuted.

### 3.2. Data Analysis

These data have been analysed using a thematic analysis approach [32]. Interview transcript and project documentation data were carefully reviewed, coded and then codes were grouped into tentative themes. Themes were then reviewed, compared and finalised, using a process similar to constant comparison in grounded theory [33].

## 4. Case Study

Add Latent Ltd. specialises in asset integrity management and maintenance optimisation with clients predominantly working in the offshore oil and gas and more broadly in the energy sectors. The company, which is part of the Add Energy group, offers a range of consulting services in all aspects of Asset Integrity Management and operational support,

 

to improve safety, reliability, regularity and compliance of operating assets such as offshore platforms, floating production storage and offloading units, drilling rigs, onshore gas plants, national infrastructure, and power plants worldwide [34].

Clients include BP, Shell, E.On Uniper, Taqa and Stena Drilling. Add Latent has traditionally had 30% growth year on year however large contracts have resulted in faster rates of growth during 2016 and 2017. The company turnover in 2016 was around £3.85M (US $4.65M).

## 4.1. Case Study Context

The case study software application draws on functionality originally provided by tools used internally by consultants in Add Latent Ltd. The first tool collects data, usually from an enterprise resource planning (ERP) system, such as SAP, regarding maintenance activity on a client site. The ERP system collects information about maintenance tasks performed, as well as any outstanding tasks. The details of maintenance tasks are then collected by an Add Latent proprietary Historical Maintenance Review Package. Maintenance tasks are categorised to enable safety-critical tasks to be prioritized. Records are kept of the time at which the maintenance task is requested through to its completion. Maintenance tasks can include repair, replacement or refurbishment of a wide range of parts used in sites such as power stations or oil and gas platforms.

Another tool, implemented using spreadsheets used the maintenance records from the Historical Maintenance Review Package to calculate key performance indicators which can be used to assess the quality of maintenance processes used at a site. Add Latent, using their experience of many different sites, can then benchmark sites against sector best practice. The range of key performance indicators enable Add Latent to identify any weaknesses in client maintenance processes and make recommendations regarding potential improvements.

Finally, the third tool uses the maintenance records and key performance indicators to produce trend information. The objective is to identify if key performance indicators are improving, or deteriorating, over time. Monitoring these trends is important for maintaining asset integrity, avoiding unexpected plant shutdowns or accidents. For example, the trends can be used to show if maintenance processes are adequately tackling a build-up of work tasks in an aging plant. A build-up of maintenance tasks might require increased investment in assess maintenance personnel or improved efficiency of maintenance processes, such as more precise targeting of resources on high risk items.

## 4.2. Feasibility Study

A feasibility study was conducted in 2012 to explore software architectures for existing tools that had been developed to support in-house staff providing consulting services. The feasibility study evaluated five architecture styles including the current standalone applications running on a client desktop PC, several different client-server configurations and a set of hosted services providing functionality to other applications.

A remote desktop implementation was experimentally evaluated but demonstrated poor response times. The initial application and the remote desktop implementation are illustrated diagrammatically in Figure 1. The remote desktop approach explored during the feasibility study revealed shortcomings in terms of performance and maintainability. There were also concerns about the ability to use remote desktop applications behind the firewalls of larger corporate clients, without special security configuration arrangements. This evaluation demonstrated the unsuitability of the remote desktop approach.

An investigation of alternatives identified a number of advantages of a cloud-hosted web application. At this stage, the benefits of a pay-as-you-go service, the potential for worldwide access and the opportunity to scale the application according to current levels of demand.

Because Add Energy's clients had desktop machines in remote locations with a wide variety of hardware specifications and installed operating systems, it was also felt attractive to use thin-client web-hosted services that shift computational demands to servers.

The decision was made to seek funding and re-architect the product into a cloud-hosted web application. A review was conducted of the available funding landscape and various potential sources of funding were identified.

## 4.3. Product Evolution

Funding was successfully obtained from Innovate UK, in 2013, under the Knowledge Transfer Partnership (KTP) scheme. The KTP scheme provides a formalised framework for university partnerships with companies and has evolved from the earlier Teaching Company Scheme (see for example [36]). A Software Engineer was employed to develop a prototype cloud hosted web application implementing functionality from the Historical Maintenance Review Package.

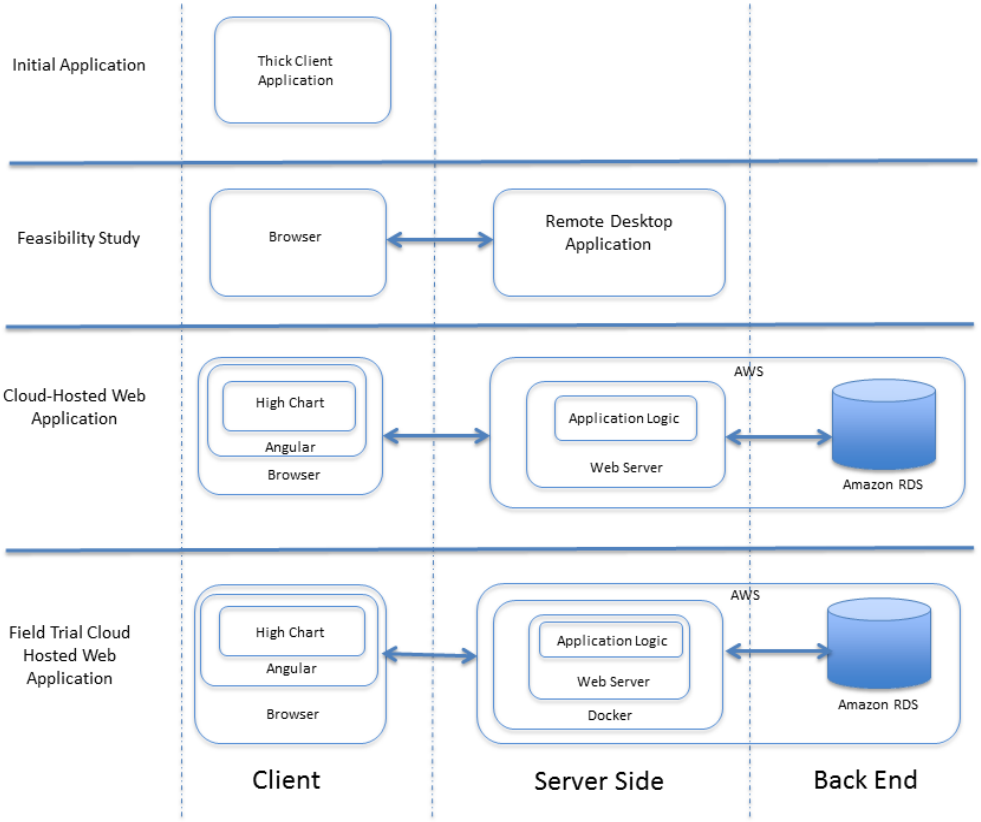


Figure 1. Architecture evolution during the innovation process

An initial prototype web application was implemented and deployed to the cloud using Amazon Web Service infrastructure-as-a-service technology. Figure 1 illustrates this evolution.

### 4.4. Minimum Viable Product

We discovered during alpha testing and internal review that the software features from the Historical Maintenance Review Package did not really form a minimum viable product. While it was useful to review the historical data, it was also important to use that data to assess the quality of the maintenance regime used by clients.

Add Latent used a set of spreadsheet formulae to calculate maintenance performance indicators. It was decided that these performance indicator calculations should be integrated into the web application. The innovation team became aware that a minimum viable product [37] demonstration was required to elicit feedback from end-users and potential clients. The minimum viable product, in this case, comprises data visualisation, using data stored in an asset database. The database is populated from information gathered using the on-site enterprise resource planning system. A series of prototypes led to the adoption of trend graphs for historical data visualisation, Aster charts to display asset type metrics and calculation of maintenance performance indicators.

An initial prototype web application was implemented and deployed to the cloud using Amazon Web Service infrastructure-as-a-service technology. Figure 1 illustrates this evolution.

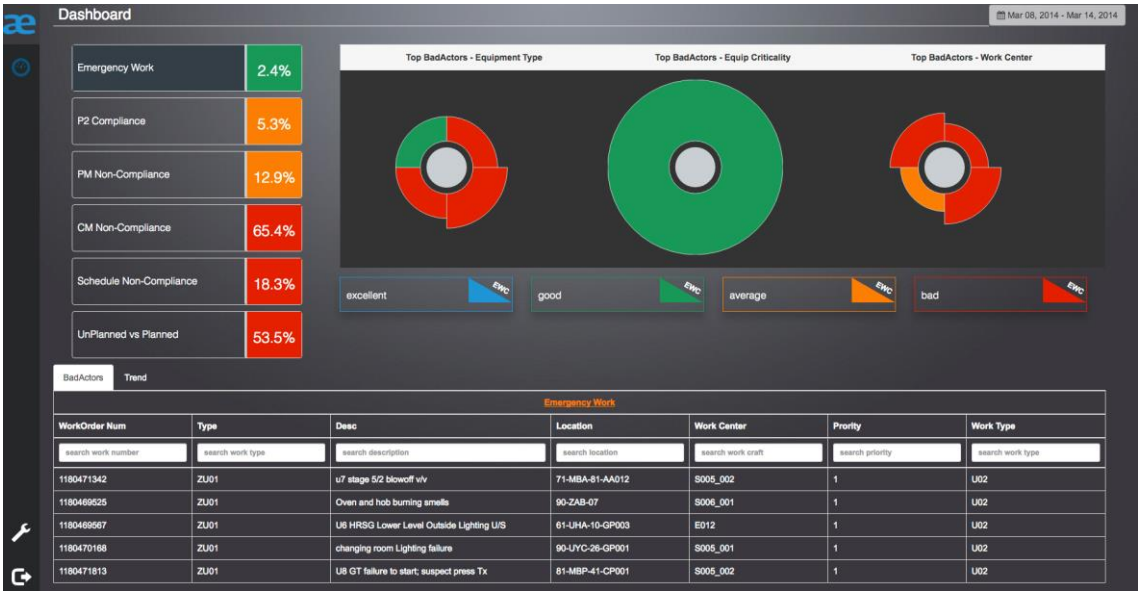


Figure 2. AimHi Field Trial Dashboard User Interface (displaying test data) [35]

## 4.4. AimHi Field Trials - Beta Testing

The new product AimHi is a web application that combines services from the Historical Maintenance Review Package with functionality from a Key Performance Indicator Analysis Suite. AimHi has been demonstrated to the maintenance engineering team at the Uniper (formerly E.On), Isle of Grain combined-cycle gas turbine power station. The power station is a lighthouse site, used by Uniper to explore best practices and use innovation to drive power station efficiency. Feedback from the field trial was then used to enhance AimHi features.

The application comprises a dashboard, shown in Figure 2. The Dashboard shows high level asset integrity data using different charts. The charts present the same data in different ways so that users can gain a full understanding of the information contain therein. For example, the aster chart gives the users visual information of each asset in various states, in relations to their critical or safety level. While the histogram chart only shows counts of assets that need to be attended to. Having various charts help engineers see the data from different perspectives and make informed decisions.

Other screens in the application display bad actor and trend data, used by asset maintenance managers to plan and target their maintenance investment to minimise downtime while also maximising the utility of spare part holdings. AimHi currently follows a three-tier architecture pattern with a user interface front end, business logic and data tier. The user interface is implemented using the Angular framework which follows model–view-view-model (MVVM) architectural pattern. This encourages loose coupling between the business logic and data tiers.

The business logic is implemented using the Spring framework with restful endpoints for consumption by other services and clients. The data tier is represented as object classes which are mapped to tables in a relational database. The whole application and its associated libraries are wrapped in a docker platform for easy of deployment in production environment i.e. the Amazon Web Services cloud platform.

## 4.4. AimHi Adoption

Following the field trials several enhancements to the AimHi product were implemented. These are summarized as:

- Revised KPI calculation algorithms,
- Enhanced dashboard user interface, and
- Additional KPI visualisations.

Discrepancies were discovered in the formulae used to calculate several asset maintenance KPIs. While there are some international standards relating to asset maintenance KPIs, it seems they are not universally adopted. Hence, it was agreed to adapt some of the KPI calculations to more closely meet the methods of calculation used in Uniper.

Enhancements to the dashboard user interface included in the areas of usability, help boxes and tooltips were added, and appearance. A specialist user interface designer was contracted to support the team to create a more professional and user-friendly appearance.

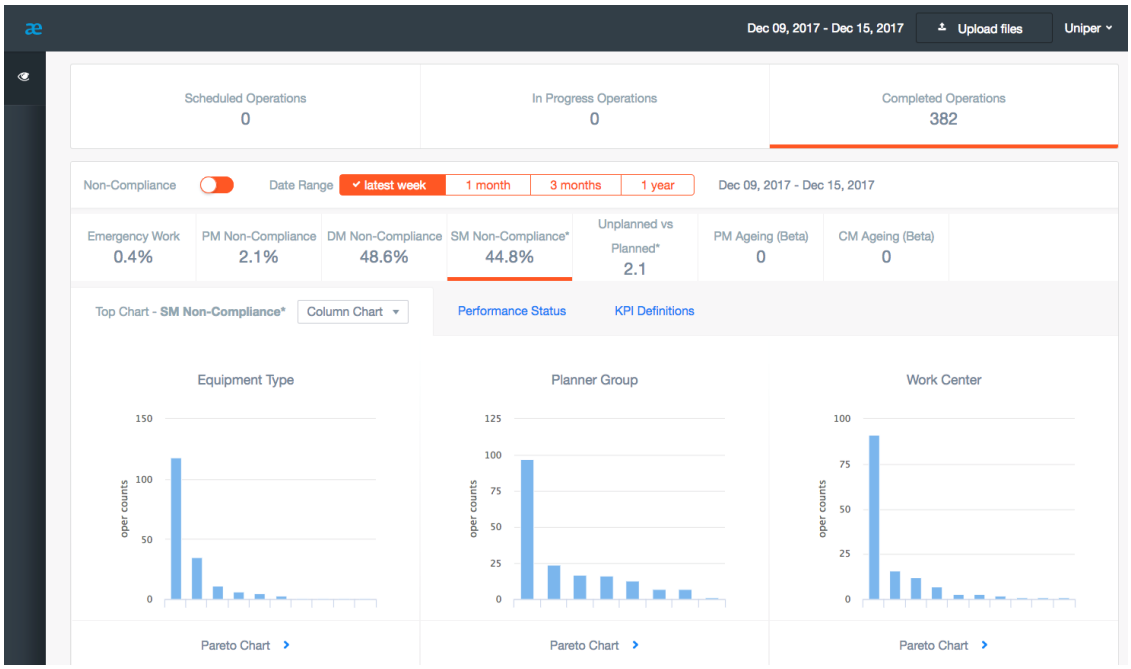


Figure 3. AimHi screen showing Pareto charts for previous week (displaying test data)

Further KPI calculations were added to enhance the functionality of the software, such as Preventive Maintenance (PM) Ageing and Corrective Maintenance (CM) Ageing. Also, additional visualization techniques were added, such as the bar charts shown in Figure 3, and the Pareto charts shown in Figure 4 and Figure 5.

At the end of the KTP project period, Innovate UK, the joint funding agency, evaluated the project and awarded the highest possible "outstanding" grade.

## 5. Case Study Findings and Discussion

The project team have successfully developed a cloud hosted web application based on functionality to support asset management in the energy and utility sectors. The cloud hosted web application has been evaluated through testing and a field trial at a major client.

The project has overcome four main challenges. Reflecting on these challenges may have wider benefit to the innovation community. The challenges are:

- KTP Associate turnover,
- Product ownership,
- Scrum process tailoring, and
- User Interface development.

Each of these challenges are now discussed in turn.

### 5.1. KTP Associate turnover

The project has benefited from a significant contribution from the current KTP Associate (the second author). Two previous Associates were employed on the project, one for 12 months another for only 3 months. Both left the project for personal reasons. The high turn-over of KTP Associates created the need to repeat induction, training and management support for each Associate. Despite some additional support being made available by the funding agency, the project capacity to focus on product innovation was reduced by Associate turn-over.

### 5.2. Product Ownership

The product ownership model used during the innovation process has ensured that the company supervisor developed a product vision and prioritised requirements to ensure the functionality built actually implemented strategic goals. The company supervisor, in their product owner role, elicited, recorded and prioritised requirements. The academic team, mentored team members in the tailored scrum process and raised awareness of agile and lean approaches.

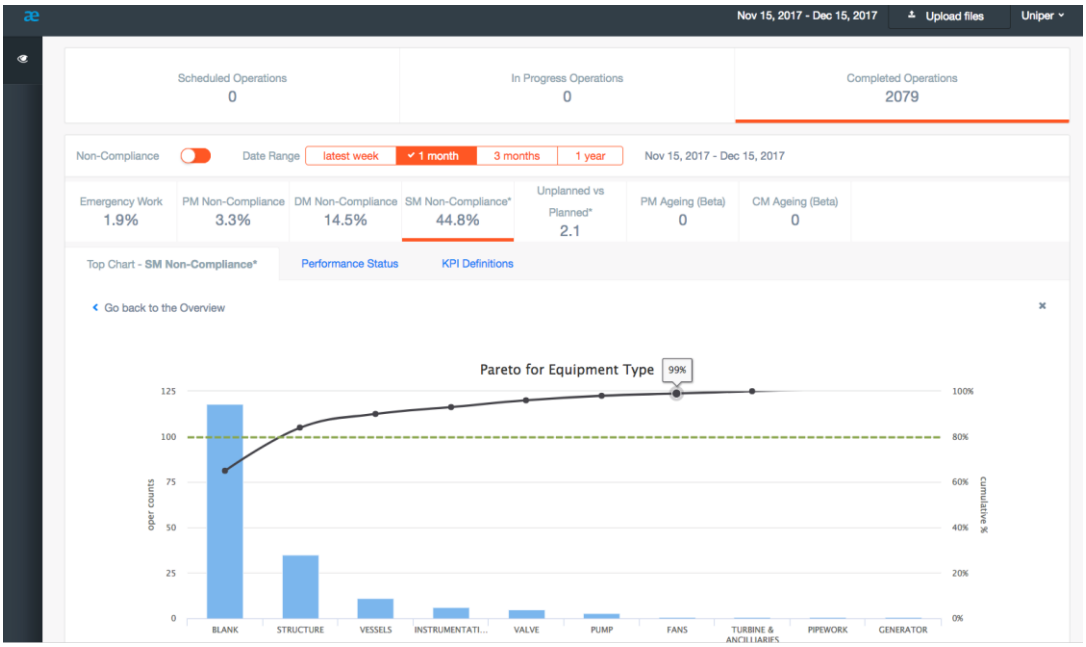


Figure 4. AimHi Screen showing KPI calculations and bar chart for previous week

However, Add Latent Ltd is categorised as a small or medium-sized enterprise and as such is typically prone to peaks and troughs in demand. Their main customer-base is in the oil and gas sector which suffered a significant downturn during 2016. The company successfully shielded the project from the downturn, despite significant financial challenges as major clients shed staff through redundancies.

During this adverse business climate, the company signed a very large contract to supply asset management consulting to BP worldwide, which meant that the main company supervisor was required to work abroad and manage a growing team to support the new contract.

The consequences of inadequate customer involvement have been well-documented [40]. This slowed the ability of the development team to obtain prioritised requirements and detailed specifications of requirement algorithms for some months while the new contract project team was put in place. The company supervisor was then able to return to focus on product ownership for the AimHi activity.

## 5.3. Scrum Process Tailoring

Our interest here is to explore the use of agile methods in a research and product innovation context, rather than for pure, conventional, software development. The development process used to develop the AimHi cloud-hosted software service is described from three perspectives: roles, ceremonies and artefacts.

**5.3.1. Tailored Scrum Roles.** The innovation team is very small comprising the development team, product owner and knowledge-base (academic) supervisor based in the university. The configuration of stakeholders in the project team is shown in Figure 6. The KTP Associate works on the company premises but is actually employed by the University.

**5.3.2. Tailored Scrum Ceremonies.** Conventional sprints lasting 2 weeks are collected into development phases lasting 2 or 3 months. Each sprint comprised a kick-of phase of product grooming, spring planning and requirements prioritization. Weekly status meetings were held with all project stakeholders meeting every month or so. Demonstrations of working code were conducted at the end of each sprint. Releases were infrequent, two or three per year, due the small size of the development team.

These development phases are interspersed with periods of requirements analysis, research and exploratory prototyping. Research was conducted on performance evaluation of multi-tenant software architectures in the AimHi product [37]. Multi-tenant software architecture enables resource sharing within a single instance running on a server. A tenant is a user or group of users that share common access and have been granted specific privileges to the software instance.

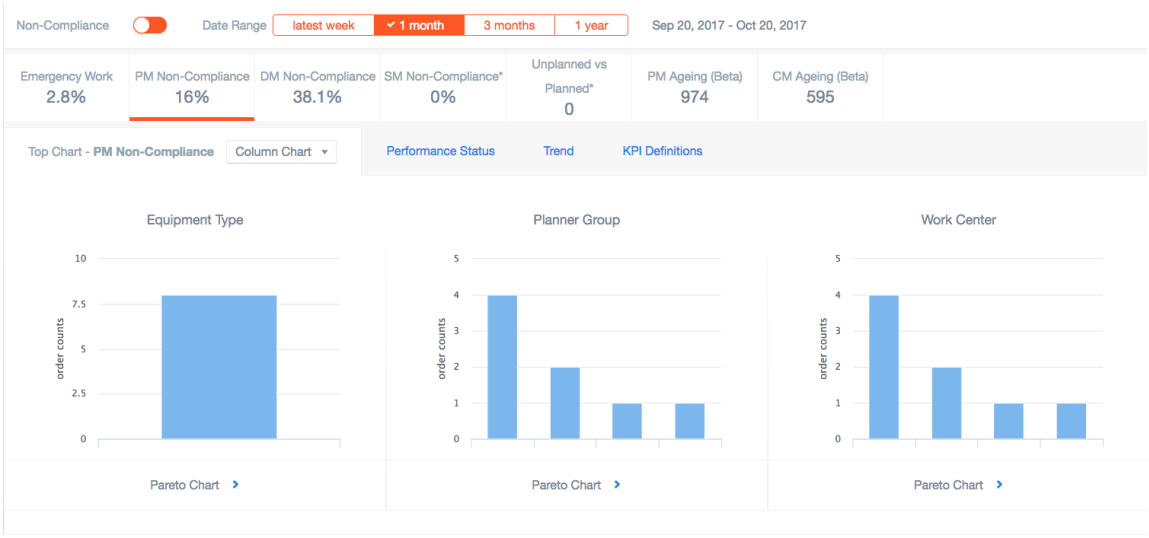


Figure 5. AimHi showing Pareto chart for equipment type based on scheduled non-compliance for 1 month

Requirements workshops were conducted by the team members with Add Latent and Uniper staff. During initial stages of requirements gathering, during 2012 and 2013, there was a wide spectrum of views expressed by stakeholders on the intended functionality of the product. The team found it difficult to create product vision at this early stage.

**5.3.3. Tailored Scrum Artefacts.** In 2015, Trello was adopted for managing requirements [39]. We found these tools useful for sharing information across distributed sites (the company premises and the university). The tool allows simple collection of backlog items. Further, files and images (such as user interfaces) can be attached to items in Trello, allowing easy adaptation of the amount of information being managed.

Some aspects of scrum do not map well to the research and innovation context. Daily stand-up meetings are not particularly useful in such a small team where such a high level of uncertainty exists.

### 5.4. User Interface Development

The project was planned on the assumption that existing products were to be migrated to the cloud. This assumption meant that the user interface for the products was already designed and satisfactory. In fact, as the functionality evolved towards a minimum viable product, it became clear the original application user interfaces were not adequate. Since, the functionality from several applications was integrated to create the minimum viable product. The User interface was developed incrementally using agile approach, as the functionalities of the application becomes clearly the user interface evolved along with it. One of the lessons learnt, the decoupling of the user interface into a component-based system helps in managing different aspects of the user interface and designing the needed aspects based on the MVP.

## 6. Conclusions

This research comprised of a 5-year innovation study involving development of a set of cloud-hosted software-as-a-service web services to support asset integrity management for the energy sector. A longitudinal embedded case study research approach has been employed. With a university employee working full-time on the company premises.

The journey from installable PC-based applications used by internal consultants in Add Latent Ltd. to the cloud-hosted application has been described. The product has been tested internally and expanded into a minimum viable product which has been field-trialed at the Uniper, Isle of Grain, combined-cycle gas turbine power station. The field trial led to further enhancements which have been incorporated into the product. A review of the enhanced product led to the adoption of AimHi by Uniper for the Isle of Grain site.

The Knowledge Transfer Partnership aspect of the project, ran from between April 2013 to October 2017, and was co-funded by Add Latent Ltd. and Innovate UK, in collaboration with the University of Salford, Manchester, UK. At the end of the KTP it was evaluated by Innovate UK and awarded the highest possible grade “Outstanding”.

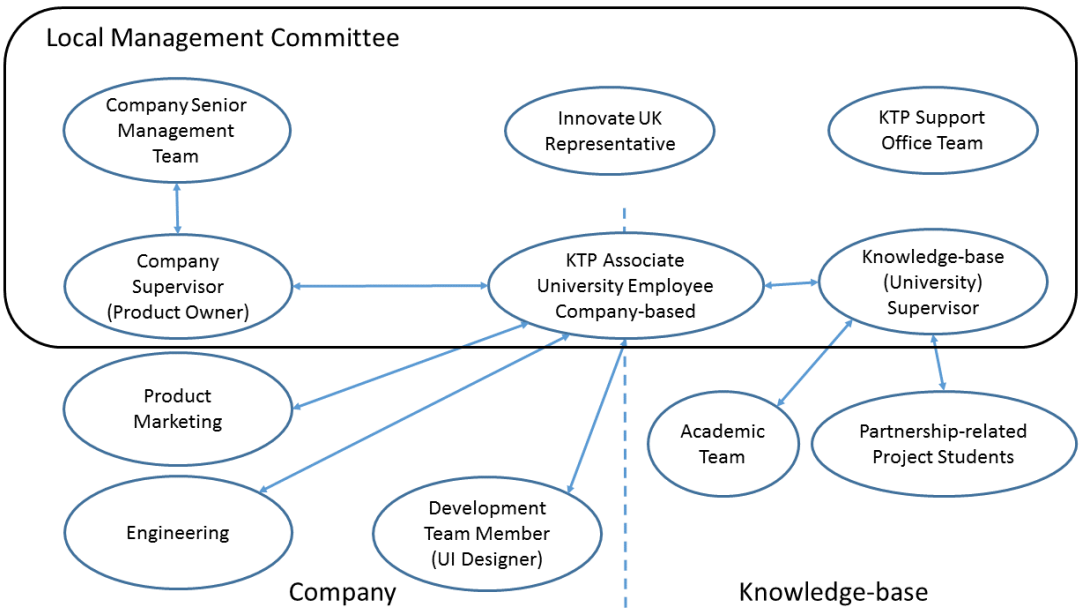


Figure 6. Deployment of Staff on KTP project

The article contributes a tailored scrum software development method for this innovation process. Conventional development sprints have been interspersed with periods of requirements elicitation, exploratory prototyping, research and reflection. The company supervisor has adopted a product owner role, defining a product vision and realising that vision by prioritising requirements. While the academic partner has mentored stakeholders in the innovation process, emphasising the need for an end-to-end minimum viable product.

In next iteration development, AimHi will be decoupled into multiple component services. Such that each component service can function as a self-contained deployable service without compromising the integrity of other services. This architecture design will follow microservice approach. The microservice will be engineered around business capacities, priorities, and governance. The modular services will communicate without other service via HTTP/Rest with JSON protocol. This is of lower complexity compared to other protocol.

Each modular service will be managed using agile methodology independently using a cross function team. Our approach is for each team to have full understanding of the service they are working on and not be burdened with the complexity of the application as a whole. This is an efficient way to manage a large, complex and multi-functional application with minimal failures.

## 7. References

[1] U.S. Chemical Safety Board, "BP America Refinery Explosion - Investigations," Mar. 2007. [Online]. Available: http://www.csb.gov/bpamerica-refinery-explosion/

[2] S. N. Ankrah, T. F. Burgess, P. Grimshaw, and N. E. Shaw, "Asking both university and industry actors about their engagement in knowledge transfer: What single-group studies of motives omit," Technovation, vol. 33, no. 23, pp. 50–65, Feb. 2013.

[3] D.-S. Oh, F. Phillips, S. Park, and E. Lee, "Innovation ecosystems: A critical examination," Technovation, vol. 54, pp. 1–6, Aug. 2016.

[4] Z. Kozhirbayev and R. O. Sinnott, "A performance comparison of container-based technologies for the Cloud," Future Generation Computer Systems, vol. 68, pp. 175–182, Mar. 2017.

[5] C. Larman and V. Basili, "Iterative and incremental development. a brief history," Computer, vol. 36, no. 6, pp. 47–56, Jun. 2003.

[6] J. Stapleton, DSDM: Dynamic Systems Development Method. Harlow, England: Addison Wesley, 1997.

[7] Alistair Cockburn, Agile Software Development: The Cooperative Game, 2nd ed. Addison Wesley, 2006.

[8] K. Beck and C. Andres, Extreme Programming Explained, 2nd ed. Boston, USA: Addison Wesley, 2004.

[9] M. Poppendieck and T. Poppendieck, Lean Software Development: An Agile Toolkit. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2003.

[10] J. W. Wilkerson, J. F. Nunamaker, and R. Mercer, "Comparing the Defect Reduction Benefits of Code Inspection and Test-Driven Development," IEEE Transactions on Software Engineering, vol. 38, no. 3, pp. 547–560, 2012.

[11] T. Mens and T. Tourwe, "A survey of software refactoring," Software Engineering, IEEE Transactions on, vol. 30, no. 2, pp. 126–139, 2004.

[12] D. Ståhl and J. Bosch, "Modeling continuous integration practice differences in industry software development,"

 

Journal of Systems and Software, vol. 87, pp. 48–59, Jan. 2014.

[13] K. M. Lui, K. C. C. Chan, and J. Nosek, "The Effect of Pairs in Program Design Tasks," IEEE Transactions on Software Engineering, vol. 34, no. 2, pp. 197–211, 2008.

[14] J. M. Bass, "Influences on Agile Practice Tailoring in Enterprise Software Development," in Agile India. Bangalore, India: IEEE, Feb. 2012, pp. 1 – 9.

[15] L. Rising and N. S. Janoff, "The Scrum software development process for small teams," IEEE Software, vol. 17, no. 4, pp. 26–32, Jul. 2000.

[16] T. Dingsøyr, S. Nerur, V. Balijepally, and N. B. Moe, "A decade of agile methodologies: Towards explaining agile software development," Journal of Systems and Software, vol. 85, no. 6, pp. 1213–1221, Jun. 2012.

[17] K. Schwaber and M. Beedle, Agile Software Development with Scrum. Upper Saddle River, NJ, USA: Prentice Hall, 2002.

[18] J. M. Bass, "How product owner teams scale agile methods to large distributed enterprises," Empirical Software Engineering, vol. 20, no. 6, pp. 1525 – 1557, 2015.

[19] J. M. Bass, "Scrum master activities: Process tailoring in large enterprise projects," in Global Software Engineering (ICGSE), 2014 IEEE 9th International Conference on, Aug 2014, pp. 6 – 15.

[20] J. Noll, M. A. Razzak, J. M. Bass, and S. Beecham, "A Study of the Scrum Master's Role," in Product-Focused Software Process Improvement, 2017, pp. 307–323.

[21] R. Hoda, J. Noble, and S. Marshall, "Self-organizing roles on agile software development teams," IEEE Transactions on Software Engineering, vol. 39, no. 3, pp. 422–444, 2013.

[22] V. Stray, D. I. K. Sjøberg, and T. Dybå, "The daily stand-up meeting: A grounded theory study," Journal of Systems and Software, vol. 114, pp. 101–124, Apr. 2016.

[23] K. Beck, J. Grenning, R. C. Martin, M. Beedle, J. Highsmith, S. Mellor, A. v. Bennekum, A. Hunt, and K. Schwaber, "Manifesto for Agile Software Development," http://agilemanifesto.org/, 2001.

[24] J. M. Bass, "Artefacts and agile method tailoring in large-scale offshore software development programmes," Information and Software Technology, vol. 75, pp. 1–16, Jul. 2016.

[25] P. Runeson, M. Höst, A. Rainer, and B. Regnell, Case Study Research in Software Engineering: Guidelines and Examples. Hoboken, NJ, USA: Wiley-Blackwell, 2012.

[26] K. M. Eisenhardt, "Building Theories from Case Study Research," The Academy of Management Review, vol. 14, no. 4, pp. 532–550, 1989.

[27] L. Mathiassen, "Collaborative practice research," Information Technology & People, vol. 15, no. 4, pp. 321–345, Dec. 2002.

[28] S. Easterbrook, J. Singer, M.-A. Storey, and D. Damian, "Selecting Empirical Methods for Software Engineering Research," in Guide to Advanced Empirical Software Engineering, Springer, London, 2008, pp. 285–311.

[29] R. K. Yin, Case Study Research: Design and Methods, 4th ed. Thousand Oaks, California: Sage Publications, Inc, 2009.

[30] E. Muller and D. Doloreux, "What we should know about knowledge-intensive business services," Technology in Society, vol. 31, no. 1, pp. 64–72, Feb. 2009.

[31] C. Robson, Real World Research, 3rd ed. Chichester, UK: John Wiley and Sons Ltd., 2011.

[32] V. Braun and V. Clarke, "Using thematic analysis in psychology," Qualitative research in psychology, vol. 3, no. 2, pp. 77–101, 2006.

[33] B. G. Glaser and A. L. Strauss, Discovery of Grounded Theory: Strategies for Qualitative Research. Chicago, IL., USA: Aldine, 1967.

[34] Add Energy Ltd, "Asset & Integrity Management." [Online]. Available: http://addenergy.no/asset-integrity-management

[35] A. O. Abdul, J. M. Bass, H. Ghavimi, and P. Adam, "Product Innovation with Scrum: A Longitudinal Case Study," in International Conference on the Information Society (iSociety), Dublin, Ireland, 2017, pp. 21–26.

[36] O. Jones and M. Craven, "Beyond the routine: innovation management and the Teaching Company Scheme," Technovation, vol. 21, no. 5, pp. 267–279, May 2001.

[37] E. Reis, The Lean Startup: How Constant Innovation Creates Radically Successful Businesses. London: Portfolio Penguin, 2011.

[38] A. O. Abdul, J. M. Bass, H. Ghavimi, and P. Adam, "A performance evaluation of multi-tenant data tier design patterns in a containerized environment," in International Conference on the Information Society (iSociety), Dublin, Ireland, 2017, pp. 115–120.

[39] "Trello." [Online]. Available: https://trello.com. [Accessed: 14-Jun-2016].

[40] R. Hoda, J. Noble, and S. Marshall, "The impact of inadequate customer involvement on self-organizing agile teams," Information and Software Technology, vol. 53, no. 5, pp. 521–534, May 2011.

## 8. Acknowledgements

This research was funded by Add Latent Ltd and Innovate UK under a Knowledge Transfer Partnership with the University of Salford, Manchester, UK.

The feasibility study was conducted by David Greenwood from University of St Andrews and funded by the Horizon High Value Products in the Cloud project with Ian Allison from Robert Gordon University and Ian Sommerville from St Andrews University. Advice from Hatem Ahriz, RGU, improved database query efficiency. Former KTP Associates Mukta Aphale (November 2013 to October 2014) and Fatemeh Raji (March 2015 to May 2015) also contributed to the project.

 