Applying the Agile Mind-Set to Service Management

Dolf J.H. van der Haven

© Powerful Answers, 2016

Applying the Agile Mind-set to Service Management

Agile is a widely used (and misused) expression that needs some clarification in order to convey its meaning in the context of Service Management. Agile is popularly associated with DevOps, Scrum, iterative working, submitting and prioritising User Stories and many other aspects that are mostly practical implementations of its core meaning. What I want to focus on is how to take the Agile *mind-set* as it was originally conceived by the early developers of Agile practices and apply this to service management, rather than take elements from the various Agile frameworks out there and try to apply those to Service Management, hoping that eventually you end up with something you can call Agile Service Management.

The Agile Mind-set

The Agile mind-set has been expressed in terms of the Agile Manifesto and its related Principles. The following is a quote from the Agile Manifesto [1] of those principles which have been slightly reworded for the context of services:

"We are uncovering better ways of providing services by doing it and helping others do it. Through this work we have come to value:

- *Individuals and interactions over processes and tools*
- *Working services over comprehensive documentation*
- *Customer collaboration over contract negotiation*
- *Responding to change over following a plan*

That is, while there is value in the items on the right [e.g. Processes and Tools]*, we value the items on the left* [e.g. Individual and Interactions] *more."*

"We follow these principles:

- *Our highest priority is to satisfy the customer through early and continual delivery of valuable services.*
- *Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.*
- *Deliver working service enhancements frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.*
- *Business people and the service provider must work together daily throughout the service lifecycle.*
- *Build services around motivated individuals.*
- *Give them the environment and support they need, and trust them to get the job done.*
- *The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.*
- *Working services are the primary measure of progress.*
- *Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.*

- *Continuous attention to technical excellence and good design enhances agility.*
- *Simplicity - the art of maximizing the amount of work not done - is essential.*
- *The best architectures, requirements, and designs emerge from self-organizing teams.*
- *At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly."*

In the project management and software development worlds, a number of methodologies, such as Scrum and eXtreme Programming (XP) have been developed that have looked to incorporate these Agile principles. The Service Management world, however, has been much less influenced by Agile, even though a number of efforts are increasingly being made, e.g. [2] and [3]. The remainder of this chapter will look into how to apply Agile principles to the Integral Service Management Framework.

Applying Agile Principles to Service Management

First of all, using practices from Agile methodologies such as Scrum in the context of Service Management does not necessarily make Service Management Agile. I have for instance seen examples where people have set up a support team using an issue backlog similar to the sprint backlog from Scrum, are doing daily stand-up meetings with that team to review progress and impediments to progress, as well as a number of other practices similar to Scrum. I am, however, not convinced if those activities make their service management practices any better. *Doing* Agile is not the same as *being* Agile, after all: the practices themselves don't necessarily lead to a more agile way of dealing with service management. What I am after is to actually *improve* Service Management practices by applying Agile principles. Practices can be defined later based on the implementation of the principles in the SMS.

One can also wonder if there is a purpose of applying Agile principles to Service Management in the first place: isn't CSI with its cyclical nature not already Agile? Does Agile contribute anything essential to Service Management or is it just a fashion statement? These questions need to be cleared up as well in order to determine the value of Agile for Service Management.

Therefore in this section I want to take a *principles*-based approach to applying Agile to Service Management, not a *practice*-based approach. The difference is that the principles give guidance on how to do things in a better way, whereas the practices will eventually follow from these principles in the specific context of the services you provide.

Agile principles, as per the Manifesto and Principles discussed earlier come down to a number of core ones when applied to Service Management:

1. Focus on *value* creation for the customer – this is the outside-in view on service management that needs to be kept in mind at all times: what is the benefit of what we do in service management for the parties that ultimately should reap its fruits, viz. the customer and end-users?
2. Close *collaboration* between the customer and the service provider – following logically from the previous item, the more the customer is involved in the service lifecycle, the more likely actual value will be created.

3. Focus on *people* – this is the internal view on service management, which puts the people and teams performing all the activities to make sure services are delivered in the centre of attention.
4. *Flexibility* in dealing with changing requirements in a changing environment.
5. *Incremental* and *iterative* service design, implementation and improvement.
6. *Simplicity* and *efficiency* in service design and operation.

I will be discussing these principles in turn, applying them to several aspects of service management.

Focus on Value Creation

All basic training in service management teaches that services should be supportive of the customer's business outcomes. Also, the very definition of a service is something that delivers value to the customer. It should therefore be self-evident that value creation is the primary objective of setting up a service management system (Agile Principle: *Our highest priority is to satisfy the customer through early and continual delivery of valuable services)*. However, one of the complaints I hear frequently is that service management systems are often too much internally-focused, meaning that by focussing on the internal workings of the processes and other aspects of the SMS, the focus on the customer is lost.

Taking an *outside-in* approach to service management rather than the traditional *inside-out* approach, we should start looking at service management from the perspective of the customer rather than start looking from the service provider's internal perspective. The former approach has the benefit that value creation for the customer is always the first priority and that the services and the management thereof need to adapt to that priority primarily and not only be based on the needs of the service provider itself.

What does this mean in practice? We can identify many areas where this moves the focus of existing practices into a new light. Take for instance Capacity Management: usually, with the inside-out approach, the design of this process is focused on measuring the usage of resources (e.g. network links, CPU, Memory, disk space) and making sure they don't cross certain thresholds. This results in often plain default reports about generic capacity use in many locations and on many service elements. Additional capacity can then be sold to the customer in case they have a continuous need of more capacity than they have today – clearly an internal focus of the service provider. However, with an outside-in approach, you first need to ask yourself what the customer expects from a Capacity Management process. Perhaps they expect an ability to dynamically adjust their needed capacity themselves where needed or even have this automatically done by the service provider when capacity is over- or underused. Or perhaps they rather have specific information about what causes a high use of capacity (e.g. a specific application or process) so they can make adjustments on their side if needed. In short, at a process-level, value creation consists of making sure that you offer the customer value using processes that are adapted to what they want to receive. This is different from offering fixed service offerings that provide features nobody is interested in, but are there because they are convenient for the service provider.

In service management, every aspect of what we do needs to be done with the aim of providing value for the customer and end-users. This needs to happen in the design, planning, implementation, operation and continual improvements of the services and of the SMS. Traditional Service Management implementations (and frameworks and standards) are often very much focused on the internal aspects of service management, as if the customer does not exist. Value is there to remind us of and let us focus on what it is exactly that we provide services for in the first place: generating value for the customers and end-users.

Customer Collaboration

The focus that Agile has on value creation leads automatically to a focus on the customer, as the customer is after all the receiver of the value created by the services. A service provider therefore is not a cookie factory that produces the same product every time. Instead, the service provider needs to listen carefully to the customer and take feedback about the provided services to heart with the aim of improving them (Agile Manifesto: *Customer collaboration over contract negotiation*).

Customer collaboration, however, goes a step further than just taking feedback to heart. In this case, the customer can play a more active role in determining the nature and shape of the services that they are willing to pay for (Agile Principle: *Business people and the service provider must work together daily throughout the service lifecycle)*. This has its limits, though: if we are talking about standard services (e.g. Cloud storage), it may cost a lot to customise them for one specific customer. Even if the customer is willing to pay for customisation, this has an impact on the service management aspects, as likely customised processes to support these services need to be defined as well. This inevitably reduces the efficiency and effectiveness of the service management system as a whole.

That said, it should be acknowledged that customers should be able to influence what a service (even a standard one) should look like: what existing service elements are deemed unnecessary and what service elements are to be added in order to provide more value. Using a Customer Advisory Board or a similar structure to actively and regularly collaborate with customers on this is a way to achieve more agility in the design of new and the improvement of existing services. The “daily” collaboration from the Agile principle may be taken with a grain of salt – the aim is to regularly interact with the customer to make sure services meet expectations.

This focus on customer collaboration also makes the role of the Business Relationship Manager (BRM) far more important and somewhat different. The BRM’s role should be more like that of a *Service Owner* on the service provider’s side, which has aspects of the *Product Owner’*s role in Agile. In Agile, a Product Owner represents the business requirements, creates and prioritises a backlog of *user stories*, which are basically feature requests for the product, and interacts with the development side (e.g. with a Scrum Master) on getting these implemented in the product. In a services environment, the BRM should be that representative of the business, providing the customer’s requests for service enhancements to the service designers and implementers in order to provide more value.

The Agile Product Owner has the responsibility to work with the customer to draw up user stories in a specific format, build a *product backlog* of user stories and then prioritise the backlog, so the service development team knows what is most important to develop. Looking at this from the services perspective, there may or may not be a case to do this, depending on the nature of the service and the way in which it has been sold. For large outsourcing deals, likely the customer cannot be bothered creating user stories and prioritising them with the Service Owner/BRM. Outsourcing means delegating that responsibility to a service provider and having it done by others. Looking at a cloud-based application or platform (SaaS or PaaS), this structure with a prioritised backlog of user stories may in fact work better: once the *Minimum Viable Service* (MVS – the basic service that provides the core valuable functions of the service; the Agile terminology is *Minimum Viable Product* or MVP) has been delivered, the users or customers can make new feature requests in the form of user stories, which get presented to the service development team as a prioritised *service backlog*. The customer will then be invoiced based on the amount of new functionality that has been added to the service. This does require the service being contracted in a way that permits fee increases based on the release of new features as requested through user stories.

Note that the service backlog would also include customer requirements related to service management aspects: this is again in line with the outside-in approach to service management mentioned before. So the customer and users have a say in how they want the service management processes to function, in particular the interaction with their own versions of those processes. Furthermore, specific demands for e.g. reporting and invoicing may be expressed through a service backlog managed by the Service Owner. Again, customisation and interaction between service management processes is only viable if the customer is large enough to warrant this. That said, also smaller customers should have a say about the effectiveness of service management processes and are entitled to an excellent service experience; perhaps not as tailored as for large customers that pay for it, but still meeting their expectations.

Focus on People

The ISMF's core premise is to extend service management with a people focus. This aspect of Agile is therefore well suited within the framework that has been described before. Agile has a few practical focus points that are worth mentioning in this context, though. These are related to the type of people to hire for a service-oriented team and the organisation of the team itself.

A Service Management implementation needs to have a focus on the individual aspects of the people performing their jobs within the framework. This is to do with a number of aspects of individual and collective effectiveness. First of all, skills are to be assessed – not only in the context of job interviews to find the right candidate, but more so to create a close team of individuals who not only have their own specialisations, but have a broader development which gives them the flexibility to deploy their activities in other areas as well. This permits them to interact more effectively across functions with other team members. Agile is strongly in favour of having staff available that can perform multiple roles if required. Netflix [4] has called these people "T-shaped" to indicate that combination of a broad background (the horizontal bar of the letter T) and a deep specialisation (the vertical bar).

As described before, skills and behaviour are directly related to the right attitude (Agile Principle: *Build services around motivated individuals)*, especially in an IT environment where (technical) specialisation often results in a lack of social engagement. Closely-working teams need people with a cooperative attitude, including openness to other people's ideas and perspectives, an active interest in trying to find new ways to achieve more value, a drive for innovation and a continual focus on improvement in general, to name a few things. Paired with the broad-and-deep skills each team member possesses, this attitude is needed to generate actual results within the team. Further aspects of attitude that Agile focuses on include a high level of trust in each other; being able to take responsibility for one's results and a high degree of adaptability to cope with change in the environment, objectives or any other aspect.

There is a belief that some Agile proponents propagate, saying that for teams to be most effective, they should be co-located. At face value, there is something to say for this, given that it is easier to communicate with people if you can walk up to them and have a conversation. However, in today's virtual workspace, where teams are more often than not virtual, viz. distributed across multiple locations, in different time zones and cultures, there simply is no way to realise this. And I see no real issue with that – today's communication tools are so varied that, when set up well, they can replace most face-to-face communication at a practical level (Agile Principle: *Give them the environment and support they need, and trust them to get the job done)*. From email to Instant Messaging, from VoIP to Tele-presence and from virtual desktop to (internal) social media, there are plenty of options to communicate. What counts is that the right medium is chosen for the right messaging. Individual coaching is of course best done face-to-face, but operational discussions can take place through any other medium.

Organisationally, a *culture* should be developed that permits teams dealing with service management to be empowered to handle issues themselves as much as possible (Agile Principle: *The best architectures, requirements, and designs emerge from self-organizing teams)*. This requires first of all a management culture that is happy to delegate and at the same time provide support to their teams where needed. This is contrary to the top-down management structure you see in more traditional companies. On the one hand, this relates to the management support that is required in ISO 20000; on the other hand, it supports the Agile idea of self-organising teams, where the decision-making power is as much as possible delegated to the teams who are running the show practically. Secondly, the *structure* of the organisation changes along with the culture: when more decision-power lies with self-organising teams, a hierarchical structure is less needed to exert control over those teams. You do need a certain level of supervisory management to make sure that the output of the various teams gets aligned with each other and with the overall vision for the company. Management is also needed for "removal of impediments" to productive working (a classic Scrum Master task) and for people management tasks such as performance management and coaching. You do not need the heavy hierarchy a lot of traditional IT companies are suffering from, though. In practice, service providers can have somewhat smaller teams organised around the services provided, where there is a cross-functional ability within the teams to support services through their design, implementation and operation phases. This also does away with the stove-piped organisations where each function is in its own tower, which makes it difficult to establish cross-functional processes that depend on a lot of handovers between towers.

Managing Change

Change is a given in life and that also applies to service providers (Agile Principle: *Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage)*. Change expresses itself in changing service requirements, changing products and services, a changing competitive landscape, changing tools and technology and many more variations of change. In fact, one of the reasons why I stopped being an engineer at some point is the fact that I got tired of having to keep up with new technologies all the time (I had developed a greater people-focus in the meantime, resulting in having to deal with change on the people side instead). In contrast to all this change in the IT services industry, service providers are often slow and rigid in adapting to change, if not plainly resistant to it. It is after all deemed safer to stick to what you know and what you are good at than having to adapt to new things all the time. Until you are out of business because nobody is interested in your services anymore, that is.

Agile, in its origin as a software-oriented mind-set, tells us to *embrace* change, simply to be adaptive to ever-changing customer needs. It is that type of change that happens most often in non-software environments as well: numerous are the cases where between contract-signature and implementation of a service the customer has changed his mind and decided they want something quite different from what was originally agreed. Network service providers, for instance, often have a hard time coping with this, simply because the nature of their business is traditionally so rigid: even simple changes such as upgrading bandwidth on an Internet circuit can be a pain that takes several weeks to implement. By extension, embracing change should also cover other types of change, such as market change, technological change, etc. Services should be developed taking a continual need for change into consideration.

How do we get this mind-set put into practice in a services environment? This can be done by making the services as flexible as possible from the start. The above network services example is in fact a good one, for new technology permits a whole lot more flexibility nowadays: paradigms such as Software-Defined Networking (SDN) and Network Function Virtualisation (NFV) cater for a lot of adaptation to changing customer requirements. This is a start at a product or services-level. Now the whole Service Management System supporting this service needs to be made flexible as well. This includes putting flexibility in everything ISO 20000 tells you to. I believe this can be solved by keeping things simple (Agile Principles: *Simplicity - the art of maximizing the amount of work not done - is essential*): simple, intuitive processes, metrics and reporting that are easy to use, straightforward to produce, well understood by customers and that can be equally effortlessly changed when needed. It is the cumbersome nature of some service management processes (and their supporting systems) I have seen that makes service providers inflexible, so if these processes are simplified, not only does the organisation become more efficient, it also provides the opportunity to change them as and when required.

Change Management

While we are on the subject of change, let's take the Change Management process as an example. There is often a kind of tension between the people running the services (i.e. the operations side) and the people wanting to change the services for the better (i.e. the

development side). The operations people want as little change as possible, given that any change is a risk for the continuity of the services. The development people want to continually enhance the services in order to improve them. The way forward is necessarily that there needs to be a middle path between the two perspectives, again from the viewpoint of value creation.

Customers usually want their change requests to be implemented both as quickly as possible and as safely as possible, i.e. without disruption to the live environment. Quick and safe only go together if a number of aspects is catered for:

1. Thoroughly developed and tested service enhancements (part of the Release and Deployment process and/or Design and Transition of New and Changed Services process in ISO 20000);
2. A regular planned release schedule – if we are working in an Agile way, iterative service enhancements (more on iterative aspects in Agile service management later) should be accompanied by an equally predictable (and frequent) release schedule in which changes are deployed that have been developed until then;
3. An efficient, simple and flexible Change Management process that does not act as a roadblock for changes, but does do the necessary (and only the necessary) checks to ensure changes can be implemented in the next release. A Change Advisory Board or similar meeting should be held frequently enough to provide this flexibility and all stakeholders need to be present to assess change requests (including an Operations representative);
4. Thorough testing of implemented changes before the release is implemented and business verification after deployment – if these are not successful (or nobody is available to do business verification), the changes need to be rolled back.

In this way, the Change Management process can turn into a flexible process that helps the company achieve a more dynamic way of implementing and improving services, thus increasing the value they provide.

Incremental and Iterative Service Design, Implementation and Improvement

Contrary to what many Agilists believe, the incremental and iterative way of working that Agile proposes is not an aim in itself, which is one of the reasons why introducing practices such as chopping work up into “iterations” or “sprints” is nothing to do with *being* Agile, it is merely *doing* Agile. The aim of iterative and incremental development is there to provide *value* much earlier in the service development process than with traditional methodologies (Agile Principles: *Our highest priority is to satisfy the customer through early and continual delivery of valuable services. Deliver working service enhancements frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale).*

So, methodologies aside, in the context of Service Management, we want to provide the customer with as much value as possible as early on in the service provisioning process as possible. Practically, this means we should adopt the Minimum Viable Service: the most basic service that still provides value to the customer. Adopting this concept permits a service provider to provide value to the customer much earlier on than when the customer has to wait

until every element of the service (e.g. including full reporting packages, nice-to-have features and other non-core aspects of the service) has been delivered. Interestingly, this also works for the benefit of the service provider, as it can then start charging the customer much earlier as well, albeit for only part of the service.

For the usual planning-design-implementation-operation-improvement service lifecycle, this means that there will be an initial release of the service that has gone through the first three phases (planning-design-implementation) and the resulting MVS will then go into operation and is available for the customer. Subsequently, this cycle is repeated for additional service elements that were not part of the initial service release until the full agreed service has been developed and moved into the operation phase. This implies a two-phased development of the service: first there is a *sequential* phase to produce the MVS and implement the overall architecture of the service. This is then followed by a *parallel* phase where incremental additions which may or may not be based on User Stories created by the customer are implemented and at the same time the possible changes needed in the core MVS to support the new requirements are implemented. This is the concept of the Dual Development methodology discussed in [5].

This also goes for the service management processes: each process is first developed as the bare Minimum Viable Process that is able to support the Minimum Viable Service. In every subsequent iteration of service development, processes may have to be further developed as well to cope with increased complexity of the service or new processes need to be added. So an incident management process that at first only needed to support a simple hosted application on one server with internal storage will have to grow along with the service complexity once it turns into a fully virtualised cloud-based Software as a Service (SaaS) offering. A Capacity Management process may wait being deployed until the MVS has reached sufficient complexity to require full Capacity Management.

Note that I see these service development cycles as initially separate from the Continual Service Improvement (CSI) process: service development is done to satisfy the agreed initial requirements for the service. CSI kicks in immediately after the MVS has gone into production to catch service improvements that have not been covered by these initial agreed requirements. However, both cycles will interact with each other in time, and their respective requirements may well end up in the same service backlog.

Continual Service Improvement

It hardly needs to be emphasised that CSI is the prime example of an iterative method to improve services. ISO 20000 primarily bases improvement efforts on the Deming Cycle (Plan-Do-Check-Act). Agile has its own version of this cycle (Plan-Develop-Evaluate-Learn) that comes down to the same principles and refers to it as “Continu*ous* Improvement” (a subtle difference that is likely only understood by native speakers and language adepts – I prefer the word “continual” to the word “continuous” as the latter does not have the cyclic nature in it that is suggested by the Deming Cycle).

Continual improvement needs to be done at both the services level and at the service management level. The service provider needs to be open to feedback both from the own

organisation and from its customers to improve services. Customer satisfaction surveys as well as internal employee satisfaction surveys need to be taken serious to use as a starting point to increase the value delivered to customers and to make the way in which services are delivered more efficient and effective. This is why I typically find the verbatim feedback on these surveys far more interesting than figures such as NPS and CLI. Verbatim feedback is far more difficult to interpret, as they may be individual issues and trends are not always easy to determine, but it does often express exactly what is bothering the customer, even if it is not given (which can be a sign that the customer or employee does not care enough to suggest improvements). It is therefore this feedback that should be used as the initial trigger to improve the services and the SMS.

Finally, it is worth mentioning that every team needs regular occasions at which it can reflect on how it is working (Agile Principle: *At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly*). This is an opportunity for self-improvement that is in line with CSI and is equally valuable in order to look objectively at the way the organisation or the team is working to provide services, learn from issues that have been encountered and improve in the future.

Simplicity and Efficiency

The final aspects of Agile to consider in the light of Service Management are simplicity and efficiency (Agile Principles: *Simplicity - the art of maximizing the amount of work not done - is essential; Working services are the primary measure of progress).* Agile inherited its attitude here from Lean, which stresses as one of its focus points the removal of *muda* or "waste." Waste is defined as anything that does not contribute to creating value.

With services, the most direct way to simplicity and efficiency is standardisation: standardisation of services, standardisation of service management, standardisation of tools, and so forth.

Standardisation of services leads to simplicity in their operation: rather than having to manage more or less customised services for each customer, you can manage the same services in the same way for each and every customer. The problem is that some customers cannot live with purely standard services and need some level of customisation. Standardisation can then go into two directions: either standardise the core service and leave add-on services up to the wish of the customers - this will at least make managing the core service easier, but requires custom management for the add-ons; or develop an extensive service that includes most if not all of the add-ons that customers may wish for – this standardises the management of the full service, even if customers don't make use of the add-on services. In fact, the latter option is against Agile principles of developing minimum viable services, as there will eventually be a lot of customers that don't use all aspects of a service. Hence, waste exists in having to support service aspects that are not used. Standardisation of a service therefore only goes so far as a well-established core service. Add-ons will be custom and will have to be so at a cost.

Standardisation of service management is important in any case: for the efficient operation of services, all service management processes must be as simple and straightforward as possible. People need to work with the processes and can do without an overly cumbersome approval

hierarchy, infrequent CAB meetings, inefficient communication structures, user-unfriendly tooling and a lack of continual improvement process. The aim is, as the Lean philosophy states, to "achieve flow in the value stream": remove all unnecessary tollgates and obstacles in the process and make resources available to provide services in the most efficient way possible [6].

This all only happens with appropriate management support. Management needs to actively support an efficient service provisioning environment and obtain buy-in from other stakeholders as well to support an efficient environment. This is why ISO 20000 focuses all the way in the beginning of the standard on management responsibility.

The Role of Documentation

A persistent myth about Agile is that its methodologies would prohibit the use of any documentation. Not only is this not true (Agile Manifesto: *Working services **over** comprehensive documentation*, not **instead of** documentation), it is also bad practice (see [5] for extensive criticism on deferring documentation in the context of software development). Coming from the software world, Agile tries to reduce overhead that is little to do with the final product and does not add value, such as extensive project plans, detailed requirement documents that are obsolete as soon as written and other non-core documentation. The focus is on creating a working application, as that is what provides value to the customer.

In the services world, this is somewhat different. The nature of services is by definition intangible and therefore needs more description than a software product in order for the customer and end users to know what they are buying. Furthermore, safeguards for the correct functioning of the service need to be agreed and documented in the form of Service-Level Agreements (SLAs) and performance targets may need to be determined in the form of KPIs. The question is, how much documentation is really needed and how extensive does it need to be?

ISO 20000 has been called an exercise in documentation rather than proper guidance on the creation of an efficient Service Management System. The current version (2011) of the standard does require the service provider to document a fair number of policies, processes and provide proof of compliance in the form of records. This is, however, the nature of a standard that organisations can certify against. The primary aim of ISO 20000 is, however, not to bog down the organisation in bureaucracy, but to provide a framework to work more efficiently providing services by requiring a (limited) number of service management elements to be put into place. Any documentation needed for e.g. an audit should be living documentation anyhow, as it is to be regularly updated based on the evolving nature of the services provided. Note, by the way, that the new version of ISO 20000-1 (expected to be published in 2018) will be lighter on documentation demands.

Back to the question: how much documentation is appropriate? I would say that in the context of services, we need the following customer-facing documentation in a lean fashion:

- A **Service Catalogue** to show customers what services they can buy;
- High-level **Service Descriptions** that clarify what business benefits a service can provide to customers (rather than exhaustive technical descriptions);

- Agreed **KPIs** and **SLAs** (if contractually required).

In terms of internal documentation, the following would be required:

- An overall **Service Architecture**, describing the overall aim, context and structure of the service. This is an architecture rather than a low-level design, so does not contain exhaustive technical details;
- Specific customer service details should be contained in a **Configuration Management Database** (CMDB) or Service Knowledge Management System (SKMS). This should also contain completed user stories, if services are being developed iteratively based on evolving customer requirements;
- The (electronic) **Service Backlog** or other means to convey the status of user stories that have been requested by the customer or the service owner.

Summary and Conclusion

The preceding description of where Agile may have a positive impact on services and service management has uncovered aspects that are worth looking at and aspects that are more difficult to realise in a services environment as opposed to a software environment. In what follows, I will summarise the main Agile principles that were discussed and will draw some conclusions of their applicability to service management.

As for focusing on value, do we need Agile for this? Not really, as it should already be recognised that value is the primary aim of delivering services. However, Agile provides a refreshing perspective where value creation and the customer's perspective on the services are central, which is lacking in some service management implementations that focus mostly on the internal activities.

It is the close collaboration with the customer that is a major contribution of Agile, also in the service provider area. The “daily” aspect of collaboration needs to be taken with a grain of salt, but the aim to more closely involve the customer in the development and delivery of services is a great way to ensure customer satisfaction is ultimately maximised.

Agile's focus on the well-being of the people who actually need to do the work is fully in line with the Integral Service Management Framework. It should be common sense to have this focus, but not all organisations have developed that far yet. Agile may well provide the push to get there.

I believe that Agile has a point in wanting to reduce unnecessary documentation, as many documents will never be read by anyone or get obsolete as soon as they have been written due to new developments and requirements. However, in a service provider environment, it is hard to cut away all documentation, simply because the service knowledge needs to be retained.

In the area of change, Agile goes a bit over board in its embrace of change as a constant (Agile Principle: *Welcome changing requirements, even late in development)*, but has a point when it comes to the need to be flexible about it. This applies to services as well as to software development, albeit in different ways. Services require a higher level of control, specifically if

they are provided to multiple customers, hence change management for (multi-tenant) services needs to be stricter than when developing a software product for a single customer.

Iterative and incremental service provisioning is an area that Agile is the great game-changer in, but it is also the area that is most difficult to apply to services. It very much depends on the type of service you are providing whether a minimum viable service can actually be developed, on top of which incremental enhancements can be regularly provided. It is also up to the customer to actually agree with this approach, where the benefit is that services should be available earlier, but subsequent enhancements are to be developed in a collaborative way. This is a departure from classic contracts, which customers need to accept.

All in all, the ISMF seems well positioned to make the move from classic, process and internally oriented Service Management, to a more externally focused, flexible way of delivering services and creating value for the customer. Once the breadth of the ISMF has been embraced, an Agile approach to service management can follow naturally.

References

[1] http://agilemanifesto.org/

[2] Gordon Groll, Jayne – The Agile Service Management Guide, 2015

[3] Pinchbeck, Lee – Agile Service Management, in: itSMF Australia Bulletin, August 2016

[4] Netflix Culture: Freedom and Responsibility, Netflix, 2012

[5] Meyer, Bertrand – Agile!: The Good, the Hype and the Ugly, Springer, 2014

[6] Kliem, Ralph M. – Managing Lean Projects, Auerbach Publications, 2015

About the Author

Dolf van der Haven was born in Muiderberg, The Netherlands, in 1971. Originally a Geophysicist, he has a wide background in IT, Telecommunications, Management, Psychotherapy and Service Management. He currently works as a Service Management Consultant at Verizon Enterprise Solutions and is Co-founder and Managing Director of Powerful Answers, a Service Management consultancy based in Bulgaria, The Netherlands and the Czech Republic. He is also member of ISO/IEC Joint Technical Committee 1, Subcommittee 40, which develops the ISO/IEC standard series 38500 (Governance of IT) and 20000 (Service Management).

Previous publications include *The Healing Elephant* (2008 in Dutch, 2009 in English), about psychotherapy; and *The Human Face of Management* (2014) about people management.

Dolf lives in Groenekan, The Netherlands, with his partner and their 75 chickens. He can be reached at dolf.vanderhaven@powerfulanswers.eu.

About Powerful Answers

ITSM Consultancy from a Human Perspective.

Powerful Answers is and ITSM consultancy that provides advice and support to companies who want to streamline their IT Service Management processes and organisation in order to provide their services in a more efficient and cost-effective way, leading to higher customer satisfaction and improved business results.

Powerful Answers is an international consultancy focussing on IT Service Management in the widest sense. We are professionals with a broad background working for large ICT companies, with deep knowledge of service provisioning processes, organisation design and international business.

What is most important to us is seeing your business improve by implementing the proper service management processes. We strongly believe that IT Service Management is the key to a successful business that so strongly depends on IT.

Powerful Answers has its base in Bulgaria, The Netherlands and in the Czech Republic, so is accustomed to working internationally, cross-culturally and in various languages.

For more information, visit www.powerfulanswers.eu.