



HyperCat



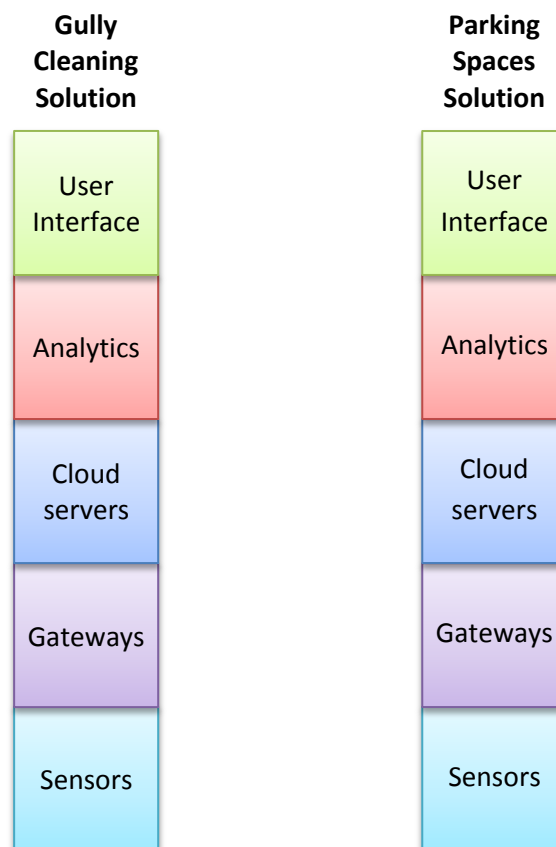
A case study

HyperCat

The “Internet of Silos”

For several years the UK’s Technology Strategy Board innovation agency had observed a growing problem which came to be known as the Internet of Silos. Connected product after connected product was being brought to market, but each was built in a vertically-integrated fashion, with all its components integrated to deliver just that one solution, without consideration for anything else.

So for example a solution to help the city manage its gulley-cleaning would have its sensors, gateways, cloud services and apps all connected together solely for managing gulley-cleaning. And a solution to help drivers find free parking spaces would have its sensors, gateways, cloud services and app all connected together solely to help drivers find parking spaces.



Whilst this approach made sense from the perspective of an individual provider, from the higher-level view of the TSB it appeared to be inefficient and indeed to be inhibiting the arrival of the true Internet of Things:

- Every solution involved a large number of parts, and a large amount of work to integrate them together. The parts and the work was largely common across all solutions and not taking advantage of this commonality meant that a lot of “reinventing the wheel” was taking place, making each solution more expensive and slower to develop than needs be.

- As multiple solutions started to appear in one domain (in our examples above, the domain is the city streets) then the managers of that domain naturally wanted to be able to integrate systems together. Towards the “edge”, they might wish to use a common fabric for connecting all sensors together, e.g. a common wireless network, or a common gateway. Towards the “centre” they might wish to integrate different services together, so that sensors feeding one service could be used by another, e.g. to stop drivers parking in a space that was about to need gulley-cleaning. The lack of any common approach made interoperability at least expensive and slow and at worst impossible when the two systems didn’t share any common way to describe their data (“semantics”).

The competition

The TSB therefore funded a competition to address this problem by demonstrating interoperability between multiple vertically-integrated solutions, thus breaking-down the silos. The competition was hugely over-subscribed and 8 clusters eventually won the bid, together comprising more than 40 companies. They were allocated over £6m to spend in 1 year, to try to crack the interoperability problem.

The companies:



Logos are trademarks or registered trademarks and are the property of their respective owners

The clusters:

- DISTANCE** (Internet of Schools Things) includes ScienceScope, Intel, Xively, Explorer HQ, Stakeholder Design, University of Birmingham’s Urban Climate Laboratory, UCL Centre for Advanced Spatial Analysis, and The Open University Department of Computing.
- EyeHub** (Security on campus) includes Flexeye, Open Data Institute, Surrey University, IBM, Guildford Borough Council
- IoTBay** (an Interoperability Hub for IoT Services), includes SH&BA, EDF, IBM, Westminster City Council, BRE



4. **i-MOVE** (Internet of Moving Objects and Vehicles Ecosystem) includes Aimes Grid Services, BT, Traak, Avanti, Placr, Merseyside Transport
5. **International Airport** (integrating Airport services) includes LivingPlanIT, London City Airport, Milligan Retail, Critical Software, AppSherpas, HWC, CrowdVision, and ECM
6. **OpenIoT** (horizontal open solutions for IoT) includes 1248.io, ARM, AlertMe, Enlight, Intellisense.io and Badger Pass
7. **Smart Streets** (City street management) includes InTouch, Carillion, BalfourBeatty, Amey, Lancaster University
8. **Stride** (Smart Transport IoT Data Ecosystem) includes BT, Aimes, Ctrl-Shift, University of Cambridge, Dartt Ltd

Working together

The TSB's goals were to a) demonstrate practical use-cases for IoT in the UK, and b) to demonstrate interoperability between clusters of companies. The project kick-off revealed a wide range of ideas and opinions. After much discussion it became clear that there could never be a single hub solution which would meet the needs of every project without becoming prohibitively complex.

We agreed a 3-month process to clarify the problem and agree how to solve it. This was a collaborative process to which everyone contributed, putting our common need for a common solution ahead of our own specific interests.

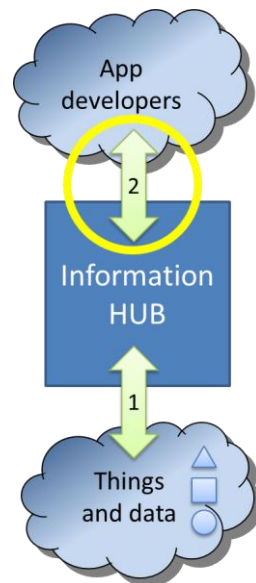
The clusters could be broadly categorised into two types:

- Vertically-focussed: Clusters with very focussed real-world use-cases, such as Airports, City streets or Transport logistics in the East of England ports
- Horizontally-focussed: Clusters trying to solve the generic problems of interoperability, regardless of application

This combination of Vertical *and* Horizontal was ideal. The risk with Vertically-focussed clusters is of building dedicated one-off integrations which are not re-usable. The risk with Horizontally-focussed clusters is of solving the interoperability problem in a theoretical way which isn't useful or practical in the real world. But the combination of Vertical and Horizontal meant that we were trying to solve problems in a generic "once-and-for-all" way, testing the solution as we went against a broad set of real-world problems.

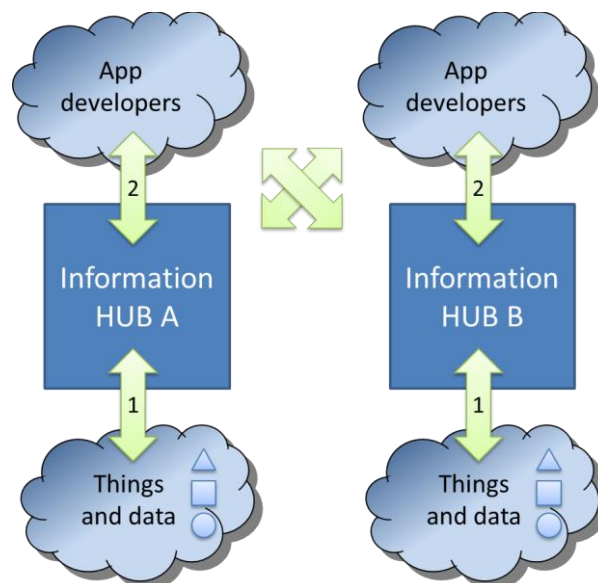
What can 40+ companies agree on?

We started by looking for commonality between the architectures of each of our platforms, and discovered that – at a very high level – they each looked something like this:



The lower-level interfaces (1) – “down” to the real-world Things – were very different from platform to platform. For example, there are many different physical networks (LANs) used to connect Things (e.g. Ethernet, ZigBee, Wireless HART, WiFi and many proprietary ones). So we judged that trying to achieve interoperability at this level would be just too hard within this project.

But the interfaces (2) “up” to the client applications were much more homogenous, because they used modern Web standards such as HTTPS, REST and JSON. This looked much more promising and so that was where we decided to focus on achieving interoperability within this project. So the goal was to make it possible for an application written for one Information Hub to be used with another Information Hub – at least as far as made sense.



Finding interoperability

Given that we already all used the same common standards on Interface 2, what was stopping interoperability? There were two problems:



- 1) Although everyone's interfaces use common standards, the way that information is *organised* varies from hub to hub. Every hub organises data in a way that makes sense to it but is not necessarily obvious to others. For example, an application seeking to find a temperature reading from each of three hubs might find it in different parts of the interface hierarchy:
 - a. A building-management service:
 - i. /country/customer/site/building/temperature
 - b. A connected home service:
 - i. /customer/home/devicetype/device/temperature
 - c. A connected streetlight service:
 - i. /localauthority/street/post/temperature
- 2) The "semantics" (the names by which things are called) vary from hub to hub, even for common concepts such as temperature. So for example:
 - a. A building-management service:
 - i. Degrees_Centigrade
 - b. A connected home service:
 - i. Temperature
 - c. A connected streetlight service:
 - i. LampTempInFarenheight

Both these are fundamental obstacles which much be overcome to achieve App-Hub interoperability. The solution in the past to making an App talk to a new Hub, has been for a human programmer to read the Hub documentation and add new code to the App. Whilst this is often not too hard, it requires a human to be present at every such new App-Hub interaction. As the numbers of Apps and Hubs explode, the number of such interactions should explode even faster (as the product of the number of each) and therefore the human programmer becomes the major obstacle.

One way to solve them would be to insist that everyone organises their data the same way and everyone uses the same semantics. However this was considered an unviable "boiling the ocean" approach, doomed to failure, because there are good *reasons* for these differences – even if they are just historical.

Therefore we needed to find some way for machines to automatically solve the problem for themselves, without involving human programmers, and accepting that differences exist between hubs' data organisation and semantics.

Discovery

This overall problem can be summarised as the problem of "Discovery". When an App talks to a Hub for the first time it needs a way to discover Resources on that hub that it can understand. If it understands only temperatures in degrees Centigrade, it needs a way to find such on a hub.

Many discovery protocols already exist and we did not want to create yet another. However as we explored existing protocols we discovered that they were bound-up in software stacks which were often complex and old-fashioned and could be used only within those stacks, which limits their general applicability. We wanted an approach which was applicable to *any* modern web interface, and built on those same web standards, so we could all adopt it.

HyperCat

As we finished the 3-month planning process we all started to contribute ideas about how this common discovery process should work. We hired an independent respected industry Chair to ensure fair play and the work was highly collaborative. 1248 CEO Pilgrim Beart facilitated the process and 1248 CTO Toby Jaffey proposed the implementation which was eventually adopted, after contributions and critique from all the other clusters. By the end of the 3-month process, we had a 1.0 specification of the standard (which later came to be known as HyperCat). 1248 now maintains and hosts the spec, and took the lead in co-ordinating the 1.1 revision which clarified some minor points.



The HyperCat standard is extremely simple – described by one participant as “the most that 40 companies could agree on”. But that is perhaps its biggest strength – it’s a very thin layer which allows apps to either explore what is available on a hub (by crawling about the interface), or search for particular types of resource outright. There is a strong, simple security model. It uses HTTPS, REST, JSON. That’s it.

All eight clusters have now implemented to this specification and have live data available. There is a rich set of tools available for creating and ingesting HyperCats, and these tools and catalogues can be increasingly being tested against each other, cross-cluster, to prove and improve interoperability.

1248 is part of the Open IoT cluster led by ARM. Within our cluster we chose to implement HyperCat on each of our five cluster-member’s hubs, making each hub a first-class citizen of the new HyperCat world, able to interoperate with hubs in other clusters directly.

During the project 1248 launched its first scalable IoT service - the **Geras** Time-Series Database - and naturally we ensured that Geras fully supports HyperCat on its interfaces. Indeed, this led to it being used as an “adapter” to conveniently bring fully HyperCat support to other data hubs. This is the subject of another of our case studies.

Conclusions

The most challenging aspect of this project was perhaps the sheer number of participants - each with their own platforms, technologies and use-cases. It would have been very easy to have become bogged-down in complexity or factionalised into different camps. That the project succeeded is a tribute to the enlightened self-interest shown by everyone, driven by a common desire to make a true Internet of Things market arrive as soon as possible by solving the “silo” problem.

The large number of use-cases, and the breadth of academic talent available to consult with, allowed us to collaboratively design a solution which is elegant in its simplicity. It does not aim to solve the problems of interoperability at every level, but simply to solve the discovery problem, enabling market forces to then drive alignment on higher-level interoperability topics such as semantics, data formats and bindings.



See Also

- The TSB's official [page](#) listing the 8 clusters
- A slideshow describing "[HyperCat in 15 minutes](#)"
- 1248's [HyperCat wiki](#) with links to Tools & Catalogues and the HyperCat specification.
- The [1248 website](#) for more Case Studies and White Papers

1248 is a team of technologists who for 20 years have designed, built and deployed connected-product platforms at scale (1 million+ end-customers per product) using the latest technologies, in areas as diverse as mobile phones, IPTV set-top-boxes and the Connected Home. Further information and more White Papers at <http://1248.io>