

# Presenting utilisation rates of resources based on positional tracking data


Alexander Kazen  
Linköping University  
Linköping, Sweden  
aleka973@student.liu.se

## ABSTRACT

Technological advancement of today provides many fields where companies may gain advantages by keeping up to date. One such field that is quite unexplored as of today is tracking the utilisation of (arbitrary) material resources. This paper evaluates the possibility to create a tool for visualising this using only simple positional tracking data as input. It shows that such a tool can be created, but also that it might need some technical improvements to be accurate in all contexts. At last, what kind of improvements might be needed are discussed.

## INTRODUCTION

### Background

With the technological advancements of today, a wave of innovations is washing over businesses with a magnitude that has not been seen since earlier industrial revolutions, according to [Westerman et al. \(2014\) \[18\]](#).  a time of such progress, companies failing to adapt and to keep their business models updated in light of the new technologies may see that competing with companies that do succeed in this gets much harder (Liu et al. 2011 [8]).

A field where such improvements and innovations is of relevance to all kinds of businesses is minimising operating costs. Slack et al. (2010) [15] argue that elimination of waste is an important principle in the optimisation of operations. They define waste as anything that is not of value to the customer, and further break it down into seven categories, two of which are waiting time and inventory.

Wernerfeldt (1984) [17] argues that a company can be seen as a bundle of resources. The idea that businesses should base their strategies on their resources rather than other factors is presented not only by Wernerfeldt, but it dates as far back as to Penrose (1959) [13]. It is also presented by Grant (1991) [5] and Barney (1991) [1] among others.

Penrose (1959) [13] argues that "at any given time, the known productive services inherent in a resource do not exhaust the full potential of the resource". Applying this argument to material resources, it ties into one of the kinds of waste presented above mentioned by Slack et al. (2010) [15]. A resource that is not fully utilised could be counted as inventory. Utilising it more means more output. However, were all resources in a given category to be utilised fully, waste elimination might once again become important, since this might mean there is waste in the form of waiting

time (there might be a need to utilise the resource even more, meaning processes will be halted waiting for the resource in question to become available).

Today, there exists a wide range of bluetooth based tracking solutions (Locatify [9], Kontakt.io [7], Estimote [3], Proxi-dyne [14] and Proximity5 [2], just to name a few). This form of tracking is a promising candidate for tracking the kind of material resources discussed ~~bove~~, since beacons are mobile and relatively cheap. ConsultingCo, a global consulting company in the IT sector, has created their own bluetooth tracking platform (hereafter referred to as simply 'the Platform') which consists of scanners (a range of devices with bluetooth capabilities) and beacons (bluetooth low energy beacons broadcasting data). The scanners listen for beacons and the system stores data on which scanner each beacon was closest to when it was last seen. This platform has in a test case been used to track the location of material resources at a hospital department.

### Purpose

Based on the given background, there seems to be a need for identifying and presenting material resource utilisation rates and thus, indirectly, costs in a given company to company management. Further, given the many bluetooth tracking platforms on the market today, a tool that calculates and presents utilisation rates based on the data available from said platforms could have a positive impact on overall company efficiency. Given that ConsultingCo already have their own Platform for which they want to explore new uses, the purpose of this thesis will be to explore whether a solution like the one mentioned above can be created and how it could be improved.

## RESEARCH QUESTIONS

With this background, this thesis aims to explore and answer the following research questions:

1. Is it possible to create a dashboard visualising utilisation rates of resources using only positional tracking data and meta data on the resources?
2. How can such a solution be affected and improved by technological advancements and improvements to the system providing the dashboard with additional data?

The biggest risk factor that might make the creation of such a dashboard impossible is that positional tracking data might not be enough to calculate the utilisation rate of said resources. Should the dashboard be deemed impossible to create, the second question takes on a larger role, answering

what kind of technological improvements would be needed for the system to be able to provide enough data to accurately calculate utilisation.

### CONSTRAINTS

The work in this thesis will use the Platform by ConsultingCo for gathering positional tracking data, rather than comparing different systems for positional tracking. This constraint is based mainly on the following two points:

1. The Platform is deemed to provide sufficiently detailed tracking data
2. Comparing the wide variety of positional tracking systems existing on the market would make the thesis grow out of scope.


### THEORY

#### Platform

The Platform is a system developed by ConsultingCo consisting of a set of BLE beacons and scanners<sup>1</sup>. The beacons regularly broadcast their unique ID while the scanners listen for all registered beacons. The scanners send information on when beacons appear or disappear from their scans to the platform server. From there, the server can in turn be configured to react in a plethora of ways, by triggering actions, contacting other services, sending responses with data to the scanners and so on.


There are two main ways the system can be configured: with static beacons or with static scanners.

#### Static beacons

This configuration is currently the main use of the Platform. When the platform is configured like this, beacons are registered in the platform and placed in static locations. For each beacon, tion or some data or a set thereof is registered. The scanners in this configuration are mobile devices, like the mobile phones of the end users. The app using the platform implements a framework handling the connection to the server. This framework downloads a manifest from the Platform server, telling it which beacons to scan for and what to do when they appear or disappear. When a certain amount of beacon data is gathered, the framework also feeds this data back to the server for history and analysis purposes.

The platform configured this way is used for making applications context and location aware.

#### Static scanners

 This is the configuration used for the work in this thesis. In this configuration, scanners (in this specific case Raspberry Pi units set up with specialised software) are installed in all locations where tracking is of value. The resources to be tracked are then fitted with beacons. The system then works in the same way as with moving scanners: the scanners send data on beacon events to the server, which in turn trigger different actions.

The use cases for this configuration differ from the ones for static beacons, since this configuration is used for tracking

<sup>1</sup>This description of the system is based on information from an anonymised oral source at ConsultingCo as well as internal presentation material

beacons instead of locating scanners. Beacons are smaller, cheaper and needs much less energy than scanners, making this configuration ideal for tracking large amounts of items.

### Data to insights

The field of Business Intelligence is defined by Loshin (2012) [10] as follows:


The processes, technologies, and tools needed to turn data into information, information into knowledge and knowledge into plans that drive profitable business action.


This work has clear ties to this field, mainly through the first part of the definition: turning data into information. The goal is to create a tool that turns data that is irrelevant to the end user (resource A was at place B at time C) into actual information in the form of utilisation rates. This process of turning data into information must be kept in mind throughout the work, to make proper reflections on what kind of data might be needed to get more accurate and relevant information.

### Similiar work

There is not much earlier work done on presenting utilisation rates of arbitrary material resources. There are however some studies on presenting more specific resources. Flix et al. (2011) [4] have studied how to visualise resource utilisation of the computing systems of CMS<sup>2</sup>. Meera and Swamy-nathan (2013) [11] explore how to create a dashboard for monitoring hardware resources in a cloud hosting service. In their proposal for a general dashboard for pediatric hospital medicine groups, Hain et al. (2012) [6] mentions resource utilisation as one relevant part of the proposed dashboard, also mentioning what to display in it.

Generalising and drawing conclusions from these studies, the following two points become obvious if they aren't already:

1. Too low resource utilisation rate is undesirable 
2. Too high resource utilisation rate is undesirable

This is basically the main point of visualising utilisation at all, and the earlier work done supports the idea that this is important to display. A utilisation rate that is too low means there are a lot of unused resources. Unused resources can be seen as inventory, and inventory means waste (Slack et al. [15]). At the same time, a utilisation rate that is too high means that a sudden spike in workload can mean, depending on the kind of resources and the specific use case, lower production efficiency, delayed completions of projects (due to waiting time, Slack et al. [15]) and so on. This in turn can lead to dissatisfied customers and lost business. These two points can be generalised and applied to the kind of arbitrary material resources relevant for this thesis. 

### METHOD

To be able to properly answer whether a dashboard such as the one mentioned in the first research question is possible to

<sup>2</sup>The Compact Muon Solenoid, one of the four collision detectors at CERN

create, the first step was to try creating said dashboard. This creation consisted mainly of two parts:

1. Creating the frontend containing the graphical visualisations
2. Creating the algorithms turning tracking data into the information needed

Even though the second question was kept in mind during the creation of the dashboard, the second step of the project was to look more closely specifically at it. Reflecting on the work done with the dashboard, an analysis was made to come to proper conclusions.

## The Dashboard

### Frontend

The very first step to creating the frontend was to decide what language and framework to use. For ease of accessibility and compatibility, it was decided that the dashboard should be a web application. Further, tooling and frameworks for creating the dashboard was to be decided. This decision was deemed not to have any real impact on the actual possibility to create the dashboard, and the decision was thus left to personal preference and experience. The decision fell on using a Javascript framework called Angular<sup>3</sup> together with a CSS framework called Materialize<sup>4</sup> and a graphing framework for Angular called ngx-charts<sup>5</sup>. For the development environment, the decision fell on Vim<sup>6</sup> for editing files and NPM<sup>7</sup> for package and dependency management.

After setting up the development environment, a basic Angular project was set up to start development in. Next, decisions on what data to present to the user were made based on the insights from the studied similar work [4][11][6]. When what data to present was clear, how it was to be presented needed to be decided. Again, insights from the aforementioned similar works were used, but to properly address the actual visualisation inspiration was also taken from user created skins [16] for a PC application called Rainmeter<sup>8</sup> often used for resource monitoring. The reasoning behind drawing inspiration from user created skins was that there is a wide variety of skins displaying the same kind of data, and it can be assumed that the users create skins displaying said data in ways they find intuitive and appealing. Using this range of skins as base, a general picture of what users overall find intuitive and clear can be made.

With design of the presentation done, the actual implementation was initiated. At this stage, since the backend was not yet implemented, mock data was used to provide the visualisations with values to display.

### Backend

When the frontend of the dashboard was implemented, it needed to be supplied with data, or rather with information based on the data available. A method for transforming data

<sup>3</sup><https://angular.io/>

<sup>4</sup><http://materializecss.com/>

<sup>5</sup><https://swimlane.gitbooks.io/ngx-charts/content/>

<sup>6</sup><http://www.vim.org/>

<sup>7</sup><https://www.npmjs.com/>

<sup>8</sup><https://www.rainmeter.net/>

in the form of timestamped events telling when a beacon appeared or disappeared in range of a scanner in the Platform into information about the utilisation of the resource associated with the beacon was needed.

ConsultingCo already had an API server connecting to the Platform to serve data from it in place, and since this is just what was needed to extract the data to be used in the dashboard, instead of creating a new API server doing the exact same thing the existing one was extended.

The API server was extended with two parts: a system for providing meta data on scanners and the actual functions for transforming the data from the Platform into the information needed. In the specific case of the Platform some meta data on the beacons already existed in the Platform itself, but this is not a necessity since this meta data could easily be stored together with the scanner meta data in the API, making this solution platform agnostic.

## Technology analysis

In addition to keeping the question "how could better technology give us more relevant data" in mind during the implementation, an analysis was made after the implementation was done. In this analysis, both pure hardware focused and more software focused advancements were considered. The analysis was done in a speculative way, looking at what could be considered the weak points of the solution and how they could be improved.

## RESULTS

The results of the work are divided into three main categories. The first of these categories is design choices, where the decisions made in the different parts of the work are presented. The second category is implementation, where the actual implementation of the dashboard is presented. The last category is improvements, presenting the results on what technological improvements might be relevant.

### Design choices

#### Frontend

The first design decision was what information to display on the dashboard. The insights from studying similar work [4][11][6] imply that the one thing most relevant to display across the different fields of resources is the actual resource utilisation rate. Because of this, it was decided that the first view presented to the user should contain a collection of all the utilisation rates in question. For this to be done, a method of displaying the utilisation rate was needed. A look through the user created rainmeter skins [16] revealed that for displaying rates in ranges (many examples were of processor or memory usage), users seemed to favour gauges. Since a gauge fills up when the value it displays is increased, it can quickly convey how much a resource is used, making it suitable for the visualisation of utilisation. To further improve visibility and to make too high or too low values stick out and catch the users attention it was decided that the colour of the gauge should be based on how far from the target value the measured value was. This highlighted the need of being able to set metadata about target values in the backend, something that was kept in mind for later.

The next decision to make was whether to display a gauge for every resource or to display a gauge for an average value for each unique type of resource. Since the amount of resources for which data is of interest at a company can become quite large and since it can also be assumed that information on what kinds of resources are sticking out in their utilisation rates is of more importance than information on each specific resource (for example, if the resource type "Meeting rooms" has a utilisation rate of 30% there is probably no need to furnish another meeting room just yet, even if Meeting Room 1 has a utilisation rate of 95%), the approach of displaying information about resource types rather than every single resource was chosen.

From time to time however, the utilisation of specific resources or the balance of utilisation between resources in a category can be of relevance. To bring back the example about the meeting rooms, if one meeting room has a utilisation rate near 100% and another is close to zero, that may be an indicator of a problem with the rarely used room. This can be applied to other kinds of resources as well, if the balance between the utilisation of the resources is off, it might indicate differences between resources that shouldn't exist. These differences could in bad cases be things like "resource A1 is more accessible than resource A2" or "resource A1 is broken and thus never used". Both these examples indicate something that probably should be addressed. Because of the possible importance of such data, it was decided that only displaying the utilisation rate of categories would not be enough. Two decisions on how to complement this were made.

First of all the three most and the three least utilised resources in each category should be displayed with the average of that category, also adding the one most used and the one least used to the gauge. For the top and bottom three resources, the possibility to compare the values easily was important. The same kind of analysis of users ways to present data [16] was done and horizontal bars were chosen as the method to visualise the balance. To further improve on the information mediated by the bars, it was decided that the colour of each bar should represent how close the value was to the target value in the same way as with the gauge.

Secondly it was decided that a detail page for each category, displaying utilisation rates of all resources in the category separately was needed. Coming back to the matter of Business Intelligence [10], while the utilisation balance was considered, it became apparent that the information provided (utilisation rate only) might not be enough but there might be more information to extract from the data at hand. Specifically: a resource with a utilisation rate close to 100% over the last month could mean one of two things: either it has been used a lot of times or it has been used for a long time in the same session. In many cases, this difference can be of interest, since the first case is something that can be affected if so desired while the latter usually can not. Since the accessible data is on where the resources have been and for how long, the information mentioned can be conveyed by displaying for how long each resource has been at each location. A way to display this kind of three dimensional data was therefore needed. An example of this is used by Flix et al. (2011) [4],

who use a heat map to visualise another kind of three dimensional data. Based on this, the decision was made to display all resources in the category in a heatmap with locations and resources and x- and y-axes and with the percentage of time spent by each resource in each location as the intensity. This heatmap will give the user a quick overview both over where each resource is used the most and where many resources are used, two types of information that can be of value when analysing the utilisation.

With this, the dashboard was deemed to contain the information needed to give users a good overview of the utilisation rates of the resources presented.

#### Backend

Since the backend was not implemented from the beginning but rather extended from the already existing API server, the design choices needed were less in number. The first design choice that was made regarding the backend was to keep all calculations for utilisation to the backend and not do them in the client, to reduce client load. Next, the decision to store all new meta data on resources and scanners in the API server instead of incorporating it into the Platform was made, to keep the solution platform agnostic.

Since two different kinds of information was needed for the dashboard (the pure utilisation rates for the gauges and the balance for the heat map), it was decided that two different routes in the REST API was preferred.

Resource utilisation is defined as "the time your resources spend working on useful projects and tasks" [12]. In the field of Human Resources, this is relatively simple to measure since employees can report how much time they spend on what they do. For hardware resources like processor or RAM utilisation in computers it is most often even easier, since the hardware reports these metrics itself. In the case of generic arbitrary material resources though, a definition of when a resource is "used" is needed. Since the only data accessible about each resource in this case was positional tracking data, this is what that definition had to be based on. To address this, three categories were defined for locations:

1. Storage
2. Usage
3. Transit

Each location in the system was then tied to one of these categories. The storage category was for locations where the resources would not count as being used (e.g. storage rooms, as the name implies). The usage category was for areas where the resources should be counted as in use. With only these two categories, it was not clear how to treat resources in areas such as corridors and other areas where resources would seldom be used but rather transported between being used and not. To remedy this, transit was added as the third category.

A resource that spends much time "in transit" is another point of information extractable from the data at hand that can be of relevance to the viewer of the dashboard. High transit times can for example mean the resource is stored too far



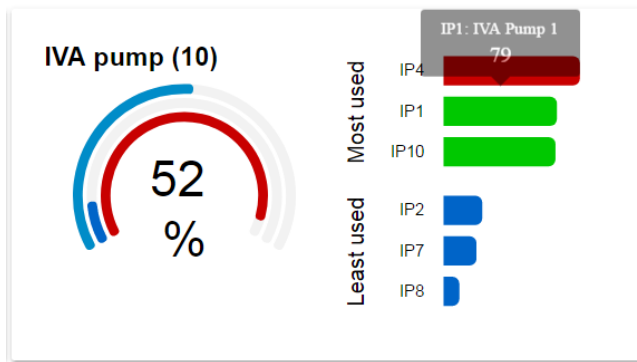


Figure 1. Dashboard card displaying utilisation rates for a given category. Values close to the target will make the graph green, while values above and below will make it red and blue respectively.

from where it is used, indicating a possible need for reorganisation. The transit time was deemed important enough to display to the user, and the decision was made that it should be displayed together with the utilisation rate of each individual resource in the category view of the dashboard.

With all the large design choices made, the implementation of the system began.

### Implementation

The system was implemented in two steps. First, the dashboard was implemented using mock data to see the visualisations, then the backend part was implemented to provide the dashboard with actual data.

#### Frontend

First of all a basic Angular project was set up using angular-cli<sup>9</sup>. When this was done, the graphs were to be implemented using the chosen graphing framework. Here the first obstacle appeared, because even though the framework had the kinds of graphs that was relevant, it lacked some features for those graphs. A quick comparison of a few other graphing frameworks for javascript made it clear that none of them had everything needed. Since the one already selected had most of what was needed and since the functionality that was missing was only small things such as the ability to set a static max value for a bar graph, it was decided that the components that were lacking functionality would be extended with what was needed.

The implementation of the parts as designed during the first phase went on without any other major road blocks. The result was a front page containing a clickable card for each resource category (see figure 1). The card contains three colour coded parallel gauges displaying the average, min and max utilisation rates from that category. The gauges are colour coded to indicate how close to the target value each of the values are. It also contains six horizontal bars, displaying the utilisation rates of the top and bottom three resources of the category, colour coded in the same way as the gauges.

Clicking on the gauge takes the user to a second view, displaying detailed information about the category. First, the

<sup>9</sup><https://cli.angular.io>

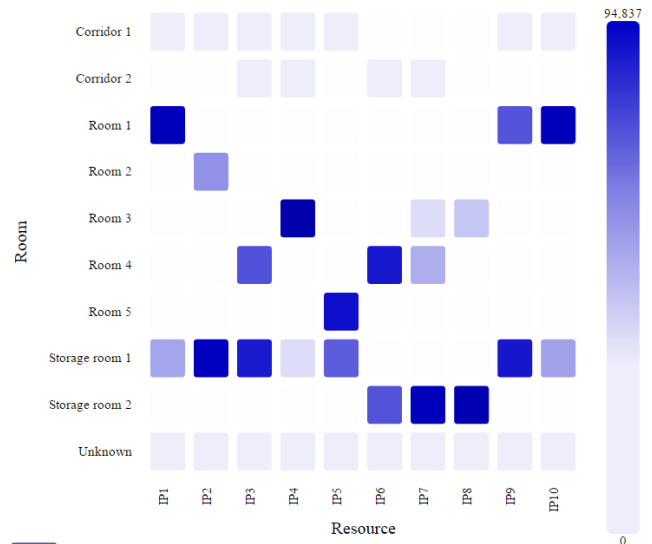


Figure 2. Heatmap displaying the balance of utilisation for all resources in a given category.

user is presented with the heatmap discussed above (see figure 2). The heatmap displays the percentage of time each resource has been in each location. Even though the scale goes from zero to the highest value in the map, the colour range is set to go from a light blue to a dark blue, instead of from white to blue. This is because with white as the colour for zero, it was almost impossible to differentiate between a really low value (for example meaning "this resource passed through here") and an actual zero (meaning "this resource has not been here").

Below the heatmap is the last part of the dashboard, the detail cards for individual resources (see figure 3). These cards contain the names of the resources along with three parallel gauges. The gauges display the percentage of time each resource was used, idle and in transit.

#### Backend

When the frontend was implemented, the backend was needed to supply it with information. Since the existing API server fetching data from the Platform was used, the basic structure was already in place and the implementation of the actual functionality could start immediately. The first thing to implement was a system for configuration. Since meta data not present in the Platform was needed, a configuration file for the API server was set up. This configuration file was implemented as a text file in JSON format specifying which category each scanner belonged to. Configuring for scanners was a Platform-specific decision, since the data available in the Platform contained references to physical scanners rather than locations as a concept. In an implementation using another tracking platform, this meta data could be set for actual locations rather than pieces of hardware. Was it not for the fact that the Platform already provided some meta data on each beacon (i.e. the type of resource the beacon was attached to), this meta data would have been included in configuration files in the same way.

With the configuration in place, the implementation of the actual algorithm began. The data from the platform is a list

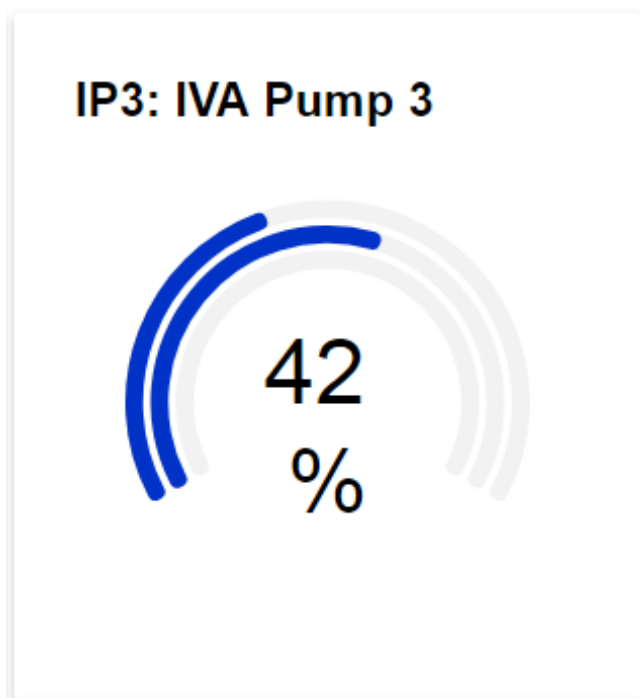


Figure 3. Dashboard card displaying percentages of time for when the resource was used, idle or in transit.

of timestamped events. Each one of these events contains the names of a scanner and a beacon. Additionally, it contains a value indicating if the event means the beacon appeared in range of the scanner, left the scanner's range or was simply lost. The API server was extended with functionality to, for each unique beacon, go through the list of all events and calculate how much of the time in the given time span was spent in each of the three defined states (used, idle or in transit). This calculation was based on the meta data from the configuration file. A resource appearing in range of a scanner would enter the state defined for that scanner, while a resource leaving a scanner would enter the "transit" state. This way, resources between scanners would always be in transit. This was not seen as a problem, as the tracking system is meant to be set up so that there are scanners in all relevant locations. A route for the REST server was then implemented fetching all events for the given time period from the Platform, running the list through the algorithm and returning a list of all beacons and the relative time they spent in each state. With all this information, the frontend could easily calculate utilisation rates, both for individual resources and for entire categories.

The first algorithm could supply the frontend with information needed for both the initial view of category summaries and the detailed views of each resource. The last piece of information needed for the heatmap was the position balance to properly display the heatmap. The algorithm for calculating this was more basic than the one calculating utilisations, since this one did not have to look anything up in configurations of meta data. In the same way as the utilisation algorithm, this algorithm goes through the list of all events for each beacon and creates a map of how large portion of the time the

resource spent in each location (in this case: seen by each scanner). A second REST route was set up to serve this data.

### Improvements

During the whole implementation, the question "could this part be improved by better technology or more data?" was kept in mind. The first major idea on how to improve the quality of the information came during the design of the algorithm. To accurately identify whether a resource is used based on only location, just translating a location to a state might not be enough. To improve this, identifying patterns in movement of beacons could be of great value. Since it is not obvious as to how these patterns would look, implementing machine learning for the ability to train the system into understanding what is being used and what is not based on advanced patterns in the locations of the resources could improve the quality and accuracy of the output information.

### DISCUSSION

The results of the implementation show that at least for a rough generalisation of utilisation, a dashboard can be created that displays useful data based on only the locations of the resources over time. One flaw of the solution developed in this work however is that all resources are treated the same way. In reality, there might be a need to configure the system so that resource type A is in use when in location 1 while type B is idle when in the same location. This problem could however be solved by adding more meta data to the configuration of the API and modifying the algorithms to take that meta data into account when calculating utilisation. Because of this, the fact that this solution does not implement this does not contradict the claim that a dashboard such as defined in the first research question can be created.

Looking further at the need for type-specific configuration as mentioned above, this is probably a great example of where machine learning as discussed in the improvements section of the results could be implemented. Letting the system learn what patterns and what locations means a certain type of resource is used could greatly reduce the need for manual configuration of the system.

Machine learning could also most probably, over time, provide a much more accurate configuration than the estimations and assumptions needed to configure the system manually could. This is of course due to the fact that the system could fine tune itself over time, but also because an implementation with machine learning could learn to recognise patterns that would not be possible to configure manually without much more complex forms of configuration.

If the constraint of "only positional tracking data" from the first research question is dismissed, even more improvements could obviously be made to the system. As mentioned in conjunction with the questions, the main risk factor of using only positional tracking data is that it might not be enough to calculate the utilisation rate. Even though this study finds the result promising for a general case, the location might very well not be enough to track utilisation in all special cases. An obvious example would be a case where resources that are in use not necessarily leaves the room in which they are stored.

The obvious way to improve the solution for such cases is to include other data than just position in the calculations.

One type of data that could easily be added without rebuilding the system as a whole would be motion data. The beacons could be fitted with accelerometers and made to broadcast the data from them as well. This data together with locational data could be transformed both information on when a resource that never leaves a certain room is being used or is idle, but it can also be transformed into more accurate information on the usage of resources that are in the current solution counted as used as soon as they are in a certain room.

Of course, even motion data would not be enough to calculate utilisation of *all* kinds of resources. Static resources that stays in place even when used need to be measures in some other way. The general conclusion from this is probable that the system cannot be fully generalised to handle completely arbitrary resources, since different types of resources will always need different types of measurements.

### FUTURE WORK

To expand on this solution and to make it as general as possible, the prospect of implementing the two major improvements discussed should be examined. Implementing machine learning could really improve the solution and extending the available measured data might be a necessity for some types of resources.

### CONCLUSION

This study finds that it is possible to create a dashboard visualising utilisation rates of generic resources based on location data, as long location can reasonably differentiate used and unused resources of the type in question. For more complex patterns than "location A means resource is used" however, more work needs to be done to properly calculate utilisation. The improvements needed to do this could both be software based (such as machine learning) and hardware based (such as including more metrics).

### REFERENCES

1. Barney, Jay. Firm resources and sustained competitive advantage. *Journal of Management*, 17(1), 1991: 99–120.
2. Bluvicion, Inc. Proximity5 - platform. Address: <https://proximity5.com/platform/>.
3. Estimote, Inc. Estimote beacons - real world context for your apps. Address: <https://estimote.com>.
4. Flix, J, Hernández, J M, and Sciabà, A. Monitoring the readiness and utilization of the distributed cms computing facilities. *Journal of Physics: Conference Series*, 331(7), 2011: 072020.
5. Grant, Robert M. The resource-based theory of competitive advantage: implications for strategy formulation. *California Management Review*, 33(3), 1991: 114–135.
6. Hain, Paul D., Daru, Jennifer, Robbins, Elizabeth, Bode, Ryan, Brands, Chad, Garber, Matthew, Gosdin, Craig, Marks, Michelle, Percelay, Jack, Terferi, Sofia, and Tobey, Donna. A proposed dashboard for pediatric hospital medicine groups. *Hospital Pediatrics*, 2(2), 2012: 59–68.
7. Kontakt.io, Inc. Your solution. beacon-enabled. Address: <https://kontakt.io>.
8. Liu, Day-Yang, Chen, Shou-Wei, and Chou, Tzu-Chuan. Resource fit in digital transformation: lessons learned from the cbc bank global e-banking project. *Management Decision*, 49(10), 2011: 1728–1742.
9. Locatify. Locatify branded apps, tour guides and games. Address: <https://locatify.com>.
10. Loshin, David. Business intelligence. The savvy managers guide. 2nd ed. Waltham: Elsevier, 2012. ISBN: 9780123858894.
11. Meera, A. and Swamynathan, S. Agent based resource monitoring system in iaas cloud environment. *Procedia Technology*, 10, 2013: 200 –207.
12. OpenAir. Calculating utilization in a services company. Address: [www.openair.com/home/OpenAirWhitePaper-CalculatingUtilization.pdf](http://www.openair.com/home/OpenAirWhitePaper-CalculatingUtilization.pdf).
13. Penrose, Edith Tilton. The theory of the growth of the firm. New York: Wiley, 1959.
14. Proxidyne, Inc. Proxidyne proxidyne sensor network - proxidyne. Address: <https://proxidyne.com/proxidyne-platform/>.
15. Slack, Nigel, Chambers, Stuart, and Johnston, Robert. Operations management. 6th ed. Harlow: Pearson Education Unlimited, 2010. ISBN: 9780273730460.
16. Various. Skins on rainmeter - deviantart. Address: <http://rainmeter.deviantart.com/gallery/23941137/Skins?offset=0>.
17. Wernerfelt, Birger. A resource-based view of the firm. *Strategic Management Journal*, 5(2), 1984: 171–180.
18. Westerman, George, Bonnet, Didier, and McAfee, Andrew. Leading digital. Turning technology into business transformation. Harvard Business Review Press, 2014. ISBN: 9781625272478.