Measurement as the Prerequisite for Automation

Hans-Christian Boos

Although computers are one of the main drivers for the automation and acceleration of almost all business processes, maintaining such computer systems is mostly manual labour.
As this seems ironic new approaches are being presented in order to make the machine take care of such support tasks itself, i.e. automatically.
Automation means that predefined actions are independently executed under specific conditions by the machine. Since carrying out specified actions like scripts and programs is the primary task of most computer systems, the challenge obviously is defining th conditions. The essence of such conditions is a set of logical rules referring to measurement data.
Yet, which values are to be collected, which relationships must be represented and how will this create an automatically maintained and documented IT landscape? These are the key questions for automation.

The Status Quo of Automation

Anyone who asks a system administrator or a respective project manager on whether they have automated their daily administrative work will usually be answered with the affirmative. This is followed by “yes, we have many jobs that work for us – if we had to do all of that by hand....”
But this shows the crux of the status quo. Today, automation is left in the hands of the technician responsible for a specific system. The focus of such an administrator it to relieve himself of repetitive tedious tasks. Anyone who has repaired the same minor item on a computer 22 times will write an appropriate script. That this results in a positive reduction of workload is indisputable.
But the actual benefit from such “private action” is usually difficult to plan. Normally it cannot be transferred to other systems or environments and it is rarely documented.
Consequently, this procedure cannot be considered automation in the conventional sense.

Although such scripts may show good results – it is certainly not possible to develop an IT strategy upon them.

The ideal process in IT support

To achieve a desirable degree of automation, first the terms of the automation environment, the results of the automation and the remaining manual labour must all be defined.

The processes in an automated environment can be described as follows:
1. A measurement process constantly monitors the correct functioning of the IT system.
2. Should a problem occur, a set of rules is activated that classifies the problem.
3. This ruleset continues to initiate actions and analyse their results in combination with the measurement data until either:

    a) The problem is resolved, or

    b) The set of rules can no longer initiate any action and passes the result of the “work done” up to that point on to an intelligent person who then attempts to solve the problem.

The steps listed above are challenging. And just starting with their execution means to determine the proper functionality of the IT systems through measurement first. That by itself is a major challenge.

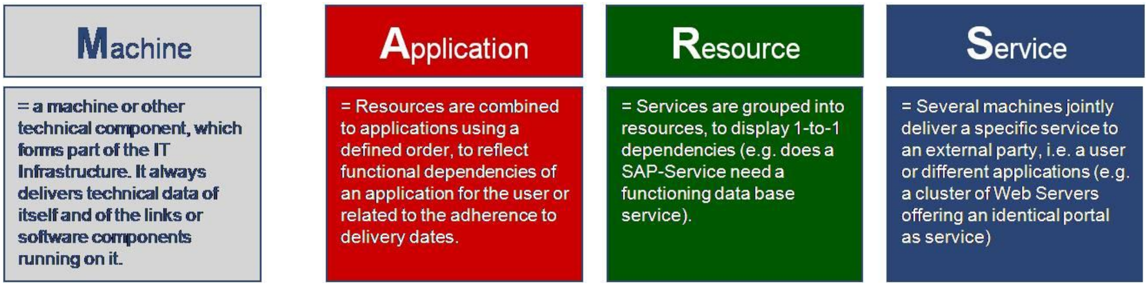
M—A-R-S: From technically closed worlds to a holistic process perspective

Conventional measurement procedures are almost exclusively concerned with technical components. It is not without reason that tools of this nature originate from the areas of “network management” or “system management”.

But reducing the functional capacity of IT to the correct functioning of the many technical components viewed in isolation falls far too short. It is much more important that the overall environment delivers the desired result for the user or business process.

The first step towards a holistic perspective of the IT landscape begins with taking an inventory of it. When this administrative act has been completed – and equipped with a process that keeps this master data in sync with reality – it is time to consider relationships and dependencies between the components. From more than ten youre of experience, arago has developed a simple and practical model for representing these relationships: The M—A-R-S model. The M—A-R-S model is based on the assumption that business processes rely on the capacity of applications to function and that hardware stands at the end of the IT “food chain”, far behind the application. Creation of the model begins with documenting which machines and applications (M—A) are employed. The next step is concerned with presenting the relationships and dependencies that exist between the machines and applications. For this purpose, it is important to identify the services that each machine – with fail-safe systems, so-called clusters – offers the outside world.
This could be a database, a web server, a virus filter or an SAP service.

The arago M-A-R-S model



Machine

= a machine or other technical component, which forms part of the IT Infrastructure. It always delivers technical data of itself and of the links or software components running on it.

Application

= Resources are combined to applications using a defined order, to reflect functional dependencies of an application for the user or related to the adherence to delivery dates.

Resource

= Services are grouped into resources, to display 1-to-1 dependencies (e.g. does a SAP-Service need a functioning data base service).

Service

= Several machines jointly deliver a specific service to an external party, i.e. a user or different applications (e.g. a cluster of Web Servers offering an identical portal as service)

These services are highly dependent on one another. For example, an SAP system would never be able to work without a database. To represent these strong connections, we combine these interdependent services into resources.
This means that the available machines, the applications, the services offered by the machines and the dependency of the services on one another are known. Now, it is simply a matter of closing the circle to the applications. I.e. extending the module to document which resources each application uses in which order.
This creates a picture of the IT landscape – represented at various levels of abstraction – that is understandable and maintainable. Not just the technical staff can understand this module, but users and managers can access the same model (in a different level of detail). Using any point in the model as a starting point, the chain of cause and effect can be used to produce statements about the consequence of a failure: Which applications are affected by a malfunction in a specific component, which services can be replaced by other

components or the scarcity of which resources leads to which consequences on the business and technical levels.

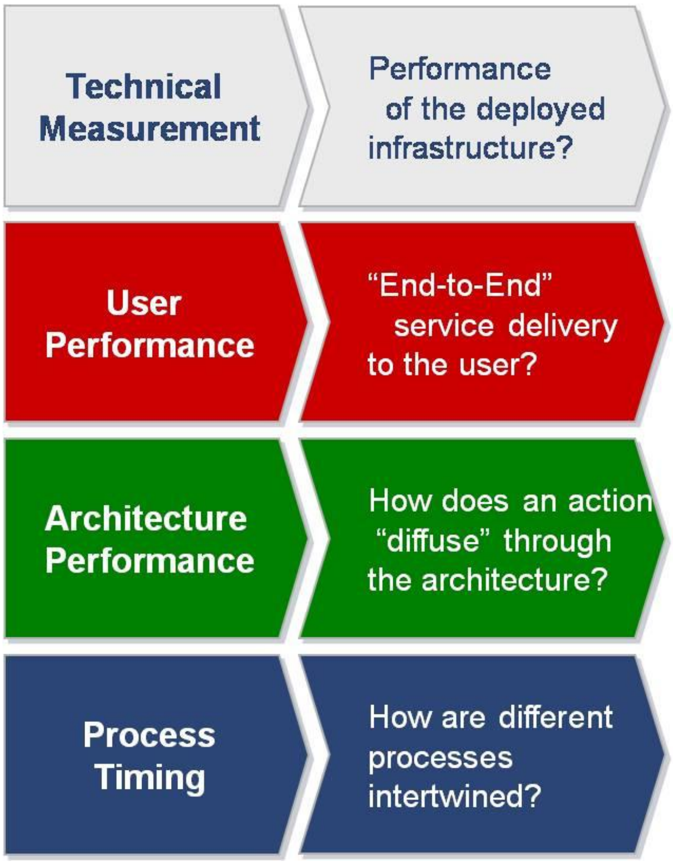
Representation of measurements and relationships

The M–A-R-S model is only a tool. Our objective is to establish a measurement procedure that monitors the functionality of the entire IT environment. The model only makes it possible to follow dependencies and draw conclusions from specific combinations of measurement results.

But what must be measured to make the potential of the recorded dependencies useable?

Classical technical measurement is not sufficient, i.e. it generates only part of the data required to automatically respond to malfunctions.

With the data that a network or system management platform collects, it is only possible to prove functionality on the machine level. As the dependencies documented by the M-A-R-S model make it possible to find the source of cascading problems in the infrastructure monitoring on the machine level is not enough.


Technical Measurement → Performance of the deployed infrastructure?

User Performance → "End-to-End" service delivery to the user?

Architecture Performance → How does an action "diffuse" through the architecture?

Process Timing → How are different processes intertwined?

Technical key performance indicators alone are insufficient

Three more groups of measurements are necessary to describe the functionality of the IT environment as a whole.

End-to-end application availability

By simulating a user on a real application it is possible to determine whether a user can access the system in reasonable time or access the system at all. This is done by the evaluation of the simulation results (true/false, response time).

Correct data processing

Through monitoring the flow of data resulting from actual and simulated use of the application, it is possible to check whether the data within the IT landscape has been processed, and if so whether it has been processed correctly. The fact that a user can use the application is not always proof of the functionality of the application. It must be proven that the data has been processed correctly and subsequently passed on (as input for other applications and processes)

Timing Data Processing

If an application's usability is given and the generated data is processed correctly, the time factor remains to be taken into account.

Some processes must be completed within a narrowly defined time limit if they are to be of any use to business. But it is also true for almost all other applications, that they are only functional when the necessary processing steps have been completed in other applications within a specific time frame. These timing periods must also be measured.

Rules – easy to understand and reusable

The four measurement groups described can record the functionality of the IT landscape itself. A ruleset – that is constantly expanded– can now recognise malfunctions in the entire environment. Through the use of the M—A-R-S model, it is possible to isolate malfunctions in specific applications, resources, services or machines. Through this process of isolation, the measured data is directed through an ever finer mesh of conditions within the set of rules. These rules trigger actions that in turn lead to the generation of new measurement data, which then finds its way into the next step of the rules – a new evaluation of the conditions or the assessment of additional conditions.

These types of rules can be easily reused because they are oriented on abstract measurement values and not customised for specific technical environments.

They are oriented on the input data of the M—A-R-S model, the master data (machine type, application type, software version, …) and the related measurement data collected in the four measurement groups. This means that the actions can be reused for all similar environments. Through the modelled dependencies, a rule is not limited to the analysis of a single technical component but can work up or down the tree of dependencies.

In this way the source of the problem is successively isolated and finally – as far as possible – automatically corrected.

There is yet another advantage to rules. The formulation of if-then conditions requires no knowledge of programming. As a ruleset, they are understandable, useable and even extendable for “normal” people.

Value Added – Measurement as a development tool

In our system operations at arago AG, we use automation not only as the product of measurement but we have gone two steps further.

On the one hand we enter the transparent presentation of measurement results, the inventory data, the dependencies from the model and the automatic documentation of the actions in a real-time reporting tool, and in doing so we offer effective and transparent support for the IT landscape. On the other hand we use the measurement data to define which skills an employee must have to be able to correct a problem that is not solved automatically by the set of automation rules.This allows us to avoid the problem of tasks being unnecessarily circulated or of individual employees “picking the best” tasks.

Like in an automated factory the tasks are assigned to the workers. Through the collation of information, the ruleset covers important preliminary work during the isolation of the problem at hand. An administrator must no longer endure mundane tasks but can finally spend his time solving the real problems.