Panel on COMP TOOLS/FUTURE COMP/BUSTECH - Tools and Applications for Service Support -
“What are Tools and Applications for Service Support – and do we really need them?”

Panelist: Andreas Hausotter

Faculty of Business and Computer Science
University of Applied Sciences and Arts, Hannover
Ricklinger Stadtweg 120
30459 Hannover
{andreas.hausotter | arne.koschel}@hs-hannover.de
A. Hausotter

- Professor at the University of Applied Sciences and Arts, Hannover, Germany
  - Faculty of Business and Computer Science
  - Department of Business Information Systems

- Teaching areas
  - Database Systems, XML Databases, Information Modeling
  - Distributed Information Systems

- Research areas
  - Service-oriented Architectures
  - Member of the CC_ITM
What is a Tool / App for Service Support?

- **Tools (in general)**
  - “A tool is any **physical item** [not belonging to the body] that can be **used to achieve a goal**.” [1]

- **Software Tools**
  - “A program that is employed in the **development, repair, or enhancement** of other programs or of hardware….
  - “It is now recognized that software tools can assist in all activities of **all phases of the software life cycle**, including management and quality assurance activities.” [2]

- **Tools (and Applications) for Service Support**
  - A program that assists in all phases of the **service life cycle**:
    - Requirements and Analysis
    - Design and Development
    - IT Operations

Fig. 1: Hand axe, Source: [1]
Fig. 2: Toolbox Source: [1]
Fig. 3: Service Life Cycle, Source: [3]
Tools and Apps for Service Support

1. Requirements & Analysis
2. Design & Development
3. IT Operations

**SOA Repository**

- Oracle SOA Management Pack
- Stardust / Infinity Process Platform
- Visual Rules
- Enterprise Architect
My Position

- Tools (and Applications) for Service Support
  - Are essential to design, develop, run and monitor services in a complex applications landscape
  - Some task may be performed automatically, e.g.
    - Test, deployment, monitoring, ...
  - Many tasks may never be performed automatically

- The more creativity required, the less the task can be automated
  - Activities in 1 - Requirements and Analysis and 2 – Design and Development require a lot of knowledge, experience and creativity
    - They are not suitable for automation
SOA Service Registry / Repository (RR)

- Functional requirements for SOA Service RR
  - Design Time
    - Service recovery and service reuse, dependency management, versioning, service classification, ...
  - Run time
    - User and rights management, life cycle management, change management, logging, monitoring and accounting, governance & compliance, dynamic service recovery, ...

- Sample provider of SOA Service RR

  - HP SOA Systinet
    - (c) HP Corp.

  - Oracle Enterprise Repository
    - (c) Oracle Corp.

  - Fusion Middleware
    - (c) Oracle Corp.

  - WebSphere software
    - WebSphere Service Registry and Repository
      - (c) IBM Corp.

  - Software AG
    - CentraSite
      - (c) Software AG
Stardust BPM Suite

Fig. 4: Stardust architecture, Source: [4]
Visual Rules BRM Suite

Fig. 4: Visual Rules architecture, Source: [6]
References

Principles and Architectures for IT Service Support Applications

Małgorzata Pańkowska
Panel on COMP TOOLS/FUTURE
COMP/BUSTECH 2017

Topic: Tools & Applications for Service Support
From Service-Oriented Analysis to Service Governance

Service Contract Design → Service Logic Design → Service Development

Service Testing ↔ Service Deployment

Serviceology
service science, management & engineering (SSME)
Principles

- laws of nature, beliefs, rules of conduct, (explicitly defined to influence behaviour, and typically based on facts and beliefs [Greefhorst & Proper, 2011]
  - scientific principles correspond to their interpretation as a law or fact of nature underlying the working of an artificial device [Meriam-Webster 2003]
  - normative principle is a declarative statement that normatively prescribes a property of something
  - design principle is a normative principle on the design of an artifact. That principle is realized by applying as part of formal analysis and design processes
- a generalized, accepted industry practice [Erl, 2008]
Proposed partitioning of architecture principles

- maximize composability
- minimize dependencies
- minimizing the availability of meta information
  - enterprise architecture
  - business architecture
  - enterprise data architecture
  - enterprise application architecture
  - enterprise technology architecture
- principles regulating the application of other principles
- principles on implementation of specific service design
- implement a standardized contract
  - implement communicative meta information
  - implement generic and reusable logic and contract
  - implement independent functional boundary and runtime environment
- generic principles
  - generic on business architecture
  - generic on data architecture
  - generic on application architecture
  - generic on technology architecture
ITIL processes

- Incident Management
- Problem Management
- Change Management
- Release Management
- Service Desk
- Service Level Management
- Financial Management
  - Capacity Management
  - IT Continuity Management
  - IT Availability Management
SLA as central point in IT service mgnt architecture

Service Level Agreement (SLA) is a part of a service contract between customer and service provider where the level of service (QoS) is formally defined.

Examples:
- Delivery time, Performance, Availability, Uptime, Speed, Accuracy, Response, Security
- Reliability
- MTBF: Mean Time Between Failure
- MTTR: Mean Time To Repair
- Response time

http://en.wikipedia.org/wiki/Service_level_agreement

[http://www.slatemplate.com/]
SLA as central point in IT service mngt architecture

[http://dx.doi.org/10.14257/ijgdc.2015.8.5.02 ]
Advent of AI chat bots for customer service

Marcelo De Barros
Principal Group Engineering Manager – Bing - Microsoft
The cost of customer service

• 72% of the businesses name improving customer experience their top priority (source: Forrester https://www.forrester.com/72+Of+Businesses+Name+Improving+Customer+Experience+Their+Top+Priority/-/E-PRE9109)

• US businesses lose a total of $62B due to poor customer service (source: Newvoicemedia https://www.newvoicemedia.com/blog/the-62-billion-customer-service-scared-away-infographic/)
Advent of AI (Artificial Intelligence)

• Industry is heavily moving into this direction (bots and personal assistants):
  • Amazon Alexa
  • Google Allo
  • Microsoft Cortana
  • Apple Siri

  • “AI and IA assist those who assist the customer. Artificial Intelligence is coming to the forefront of how a company creates a better CX. Machines’ ability to interact with humans is stronger than ever. AI will help us make better business decisions, many of them positively impacting the customer. AI won’t necessarily take over the human function, although in some places it can and will, but it will assist customer support people, becoming an IA, or Intelligent Assistant.”
  • “Chatbots are getting better. This ties into AI. The online text conversation we have with a company’s customer support center may not be with a live person, but a computer. When a machine can create a positive experience for the customer, everyone wins. The best chatbots are able to not only respond to requests and questions, but also recognize when the customer is confused and seamlessly hand off the conversation to a live customer support rep.”
And that’s already happening

- **Taco Bell**: ordering Tacos via chat bots
- **UPS**: check the status of UPS packages via Facebook Messenger Bots
- **Staples**: ordering office supplies with AI bots
- **Wynn hotels**: querying for hotel and room information using Bots
- **Comcast’s Trim Chat Bot**
## General Chat Bots vs. Customer Chat Bots

<table>
<thead>
<tr>
<th></th>
<th>General Chat Bots</th>
<th>Customer Chat Bots</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Notification</strong></td>
<td>re-engaging users</td>
<td>Might be interesting in a hybrid scenario</td>
</tr>
<tr>
<td><strong>Structure</strong></td>
<td>much less NLP, more quick replies</td>
<td>NLP becomes more appealing (intent discovery)</td>
</tr>
<tr>
<td><strong>Contextual</strong></td>
<td>location-aware, subject-aware, personal</td>
<td>Same principle applies</td>
</tr>
<tr>
<td><strong>Social</strong></td>
<td>shareable, embedded into H2H conversation</td>
<td>Private</td>
</tr>
<tr>
<td><strong>Fundamentals</strong></td>
<td>blazingly fast, platform-agnostic</td>
<td>Same principle applies</td>
</tr>
<tr>
<td><strong>Purpose</strong></td>
<td>created with one purpose</td>
<td>Multi-purpose but with quick funneling</td>
</tr>
<tr>
<td><strong>Autonomous</strong></td>
<td>no human intervention</td>
<td>Hybrid with reduction in human resources</td>
</tr>
</tbody>
</table>

Bots will replace people before they replace apps \(\text{source: VentureBeat}\)

http://venturebeat.com/2017/01/16/bots-will-replace-people-before-they-replace-apps/
But it will take some time...

• What do you think? Are Chat Bots the future of Customer Service?
Microservices for Business Applications - Future directions, challenges, and limitations

Panel COMP TOOLS / FUTURE COMP / BUSTECH
”Tools and Applications for Service Support”

Athens, Greece
February, 21st 2017

Prof. Dr. Sascha Alda

Department of Computer Science
Bonn-Rhine-Sieg University of Applied Sciences
Sankt Augustin, Germany
Short introduction to Prof. Dr. Sascha Alda

Professor for **Computer Science**, Bonn-Rhein-Sieg University of Applied Sciences (BRSU)

Contact:
E-Mail: sascha.alda@h-brs.de
Web: sascha-alda.de

**Background:**
- Main areas of interests: software engineering, software architecture

**A short CV:**
- Three years of industrial experience (IBM and Accenture)
- Doctoral degree, University of Bonn, Germany (2006)
- Diploma in computer science, University of Koblenz, Germany (2000)
Conway’s Law (Conway, 1968)

„Organizations which design systems […] are constrained to produce designs which are copies of the communication structures of these organizations“

One Example: Technical decomposition of teams

UI Team

Backend Team

DB Team

User Interface

Backend (Business Logic)

Database

New feature:
e.g. a new UI-Element

Communication in all three teams!! Eventually: bad design due to (bad) communication overhead
Conway’s Law (Conway, 1968) as an Enabler for good design

“Organizations which design systems […] are constrained to produce designs which are copies of the communication structures of these organizations“

Alternative: **Functional decomposition of teams**

Team Product Search

- User Interface
- Backend (Business Logic)
- Database

Team Order Process

- User Interface
- Backend (Business Logic)
- Database

New feature:
e.g. a new UI-Element

Communication in one teams, only!
Better design, when no dependencies among functions are given
A Microservice indicates an architectural style (..), in which the software system is decomposed into functional modules, so called Microservices. (Starke, 2015)

- Each Microservice has its own runtime environment and, thus, runs independent to other Microservices.
- A Microservice has its own domain model (Bounded Context) (Evans, 2004) (Wolff, 2016)
- Communication with other Microservices across network (REST, http)
- Flexible deployment with thin virtual machines (e.g., Docker)
- Size: Nanoservice (some 100 LoC) vs. self-contained Microservice (with own UI) (Wolff, 2016)
Future Directions (Arguments in favor of Microservices)

- **Team Product Search**
  - User Interface
  - Domain Model
  - Backend (Business Logic)
  - Database

**Runtime Environment**
- **Microservice**
- **Virtual Machine (Docker)**
  - `bookFlight() : Status`

**Strong method for the flexible adaptation of software architectures even at runtime**

**Reduction of time-to-market leads to cost-reductions**

- Strong when combined with tools from Continuous Delivery: Development and Operation can be automated to a great extend (**DevOps**)

- Based on both modern technologies, tools, and appreciated methods from Software Engineering

- Yet, mostly applied in big digital Internet businesses (e.g. Netflix, Amazon) ….

**Adoption of Microservices to (On-Premise, Cloud) Business Applications will expose many benefits!**
**Problems and Limitations**

- **Team Product Search**
  - User Interface
  - Domain Model
  - Backend (Business Logic)
  - Database
  - Runtime Environment
  - Microservice
  - REST
  - bookFlight() : Status

**Flexible deployment of (too) many Microservices:**
- No software architecture can be maintained in a long run

**Communications among (too many) Nanoservices**
- Breaks Martin Fowler’s First Law (Fowler, 2003):
  - **First Law**
    - My First Law of Distributed Object Design: Don’t distribute your objects (From P of EAA).

**Independent modules and domain models:**
- Monitoring of KPIs of the whole systems will be challenging

**Flexible deployment and independent modules:**
- Hard to implement in application scenarios with long running transactions (e.g. insurance domain)
References


Thank you!!

So, what’s your opinion about Microservices? What about effective tool support for building and running Microservices?

Your opinion Matters
Security Risks with Open Sources

Woomin Hwang
National Security Research Institute
South Korea
Open Sources Everywhere

• “There are people out there running open source web frameworks on open source server frameworks, with open source SSL stacks on open source web servers, sitting in an open source container, running on an open source kernel in an open source hypervisor.”*

- Nicko van Someren, Linux Foundation CTO

* “Security is the biggest bug of open source, says Linux® Foundation CTO,” by Roland-Moore Colyer, the Inquirer, April 13, 2016
Vulnerabilities of Tools affect entire system

• Robert O'Callahan
  • Former Mozilla developer
  • “now that I've left Mozilla for a while, it's safe for me to say: antivirus software vendors are terrible; don't buy antivirus software, and uninstall it if you already have it (except, on Windows, for Microsoft's)”

• Of course not limited to open source softwares
• Indicates increasing risk of vulnerable tools

http://robert.ocallahan.org/2017/01/disable-your-antivirus-software-except.html
For example,*

<table>
<thead>
<tr>
<th>CVE Number</th>
<th>Open Source Target /Affected Softwares</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CVE-2015-7547</td>
<td>all Linux servers/web frameworks /API web services which use the GNU C library.</td>
<td>Enabled hackers to compromise apps via a man-in-the-middle attack</td>
</tr>
<tr>
<td>CVE-2016-5340</td>
<td>Android</td>
<td>After the malware’s installation, the attacker could gain root access to the device. This put all system contents and controls (including sensitive data, microphone, GPS and system changes) at risk of exploitation.</td>
</tr>
<tr>
<td>CVE-2016-6662</td>
<td>Oracle’s MySQL DB (5.7.15, 5.6.33 and 5.5.52), MariaDB and PerconaDB (clones)</td>
<td>By injecting malicious settings into MySQL configuration files, it allowed attackers to gain full access to the server on which the affected MySQL was running. This meant hackers could view/change/erase any entries they wished.</td>
</tr>
<tr>
<td>CVE-2016-0636</td>
<td>OpenJDK (Oracle Java SE 7 Update 97, and 8 Update 73 and 74 for Windows/Solaris /Linux/Mac OS X)</td>
<td>Could be remotely exploited without any need for authentication details, such as passwords or usernames.</td>
</tr>
<tr>
<td>CVE-2016-0728</td>
<td>Android OS, Linux 3.8 and higher</td>
<td>The Linux bug had been around since 2012</td>
</tr>
<tr>
<td>CVE-2016-5696</td>
<td>Android 4.4 or later, as well all Linux OS running version 4.6 and earlier</td>
<td>Exploiting a weakness in the TCP of all relevant systems. It enabled the attacker to degrade the privacy of anonymous networks (e.g. Tor browser), track users’ online activity, hijack a conversation between hosts and terminate a conversation.</td>
</tr>
</tbody>
</table>

* : Selected CVEs and desc from https://www.whitesourcesoftware.com/whitesource-blog/open-source-security-vulnerability/
Is Open Source good for security?
‘Many Eyes’ theory to open sources

• “The enemy knows the system”

• “A cryptosystem should remain secure even if everything about it other than the key is public knowledge”

• Linus Torvalds - “Given enough eyeballs, all bugs are shallow”

Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix will be obvious to someone.
Supporters say,

• More peoples can look at source codes
  • With various technical backgrounds
  • More enthusiastic experts can be involved

• ‘many eyes’ enforces contributors write more clear code, adhere standards, ...

• Fast feedback
  • Reported important vulnerabilities in an open source project is patched within a day or two
Opponents say,

• Simply being ‘open’ source is not enough

• Also opened to hostile eyes
  • Helps reverse engineering
  • Learn from the open source
    • E.g.) Learn from Linux vulnerabilities, try it to Windows

• Only popular open sources are inspected by the ‘many eyes’
  • The number of contributors/involved people
  • Expert eyes are better than random ones
    • Technical/Technological level of participants
  • Source code coverage of ‘volunteer inspectors’ effort

• Lazy Feedback
  • Many projects are left unmanaged for a long time
So, what do you think about this?