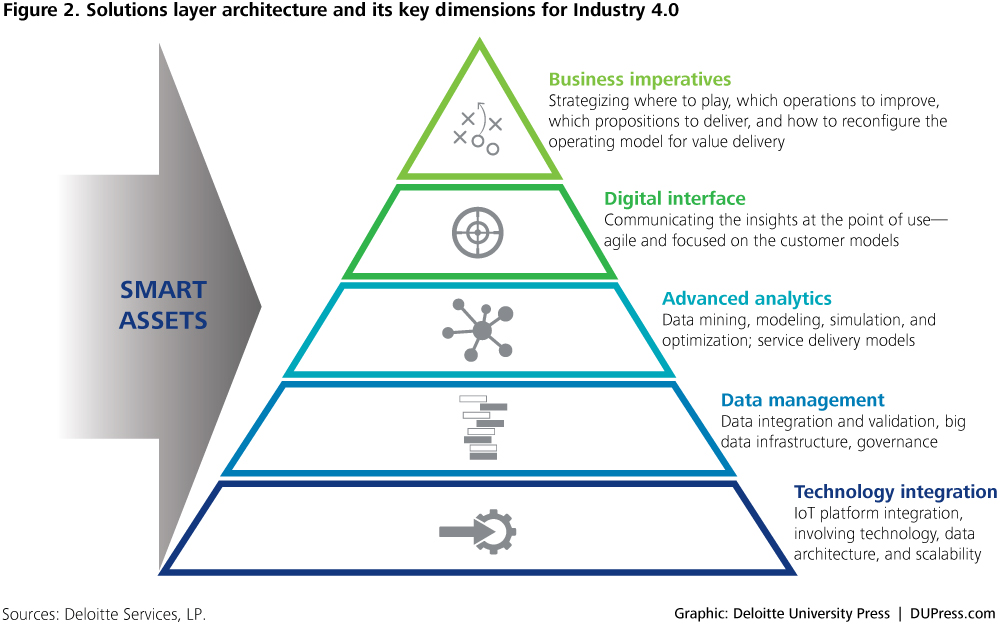
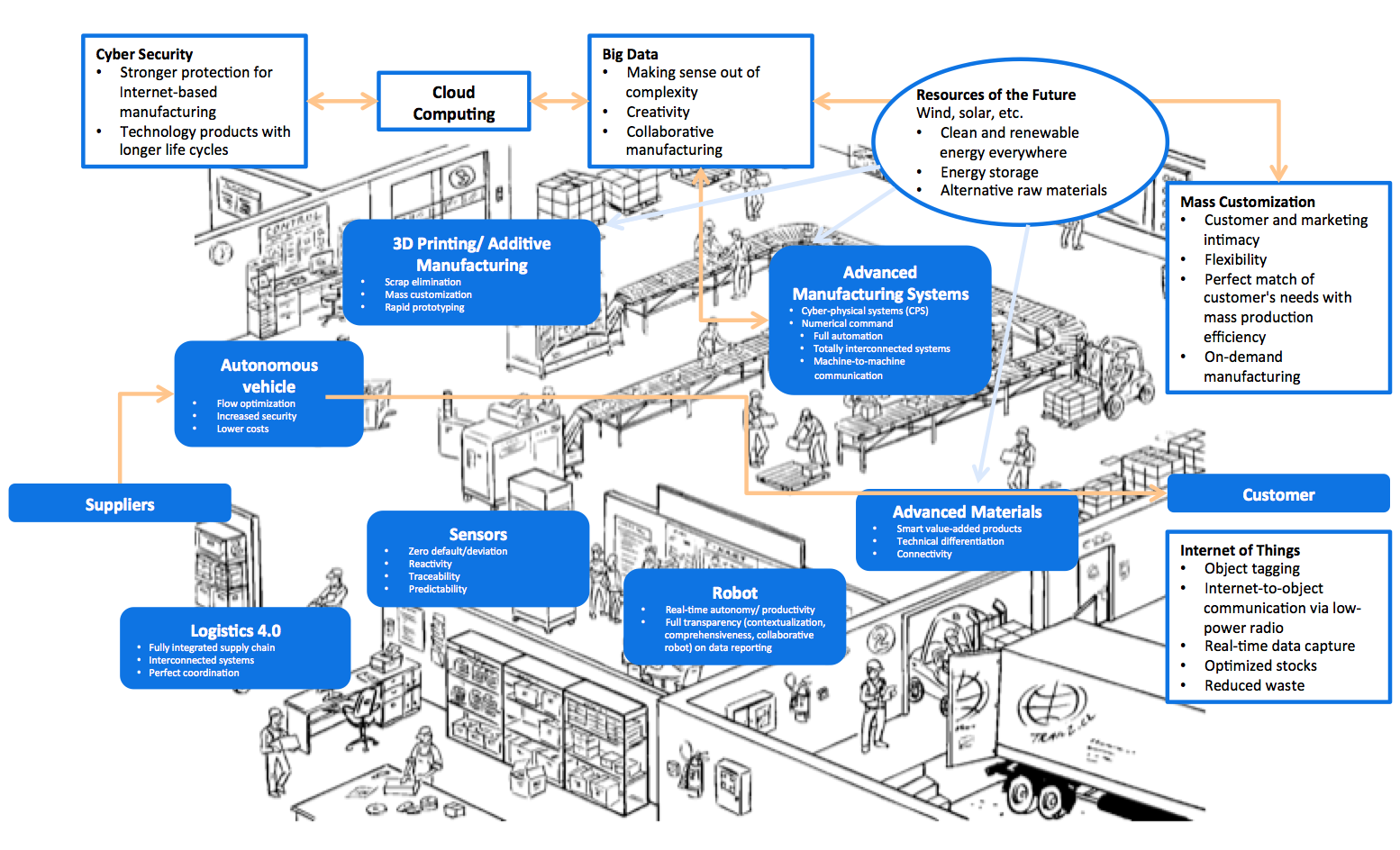
# Industry 4.0 made easy – Apps instead of expensive programming (ver pdf en desktop)

* **Lean Industry 4.0 Implementation Model = Intelligente Sensoren/Aktoren + dezentrale wireless field communication (OPC UA compliant, soft-router?) + Web App + Cloud Database Sytem** (dezentralized intelligence and distributed visualization and control? (through web app))





Mehrheit der KMUs haben die 3.0 Industrielle revolution nicht erlebt bzw. sind nicht vorbereitet für eine Traditionelle, zentralle Automatisierung.

**🡪 Übersprung der 3. Industrielle Revolution direkt auf Industrie 4.0**

# IDEA 1

**Dynamische Visualisierung für verfahrenstechnischer Anlagen**

Ich würde bevorzugen, diese Aufgabe für fertigungstechnischer Applikationen zu machen, anstatt für verfahrenstechnische Applikationen.

Was mich am besten interessiert ist das Automatisierung von Fertigung, vor Allem wegen meinem Vergnügen für Roboter (Robot Arms, Additive Fertigung inklusiv 3D Druck, sowie subtraktive, klassische Fertigung (mein Vater hat davon eine Firma). Es ist kein Zufall, dass ich Mechatronik studiere, und dass ich ein Master bei Robotics, Kognition and Intelligence später absolvieren will, und dass ich mich schon mit Robotprogrammierung, 3D Druck, Web-Entwicklung in meinem Studium aber noch mehr in meiner Freizeit auseinandergesetzt habe.

Ich weiß, dass solche GUIs für verfahrenstechnische Applikationen weiterverbreitet sind, hauptsächlich weil sie kontinuierliche Prozesse sind.

Nichtdestottrotz ist es natürlich auch möglich diskrete, fertigungstechnische Prozesse zu modellieren und über sie als eine Visualisierungsmodulbibliothek verfügen zu können.

Was die dynamischen Animationen der einzelnen Systemkomponente (der Library) betrifft: diskrete fertigungstechnische Systemelemente können vielleicht nicht so gut wie die von der Verfahrenstechnik animiert werden (bei Verfahrenstechnik: Füllstand, Farben für die verschiedenen Stoffe laut Standard z.B. Orange für Säuren, usw. , Leitungen sind auch vorhanden, die direkten Verbindungen zwischen den einzelnen Systemelementen erlauben und gleichzeitig das „Materialfluss“ des Systems visuell darzustellen.

Für diese Analyse braucht man Datenbanken mit Mock-Data bzw. Real-Data. Die Integration mit der Legato Datenbanken wurde uns erlauben, das Real-Data zugreifen zu können.

Die oben genannten Ideen sind nur Vorschläge. Ich bin mir sicher, dass mit der korrekten Betreung, diese geleitet werden könnenn, um was Interessant für

Manufacturing methods: Batch production, Job production, **Flow production**

Eine meiner Wahlfächer, was mich auch sehr gut gefallen hat ist „Materialfluss und Logistik“. Deswegen kenne ich mögliche Materialflussanalyse, die sehr interessant für diese Anwendungen wären (z.B. Warteschlangen Theory, Sankey-Diagrammen, Bottleneck-Analyse, aus Dreieckrasterverfahren Groblayout und dann Blocklayout der Fertigungslinie, unter anderen …). In dieser Weise, könnte man nicht nur das System in Betriebszeit visualisieren, sondern schon bei dem Aufbau des Systems, Materialfluss- und Logistikanalyse zu erschaffen.

Aus den oben erläuterten Gründen, schlage ich vor, das Thema ein bisschen zu modifizieren:

Dynamische Visualisierung für verfahrenstechnischer Anlagen 🡪

**Dynamische Visualisierung für fertigungstechnische Anlagen mit Funktionen zur Überwachung und Analyse von Materialfluss**

# IDEA 2

**PWA with Drag & Drop industrial system modules/components (web component) for Creation, Monitoring and Control of Industrial Processes 🡪 flexible webSCADA GUI constructor:**

1. **Industrial process MODELS in SVG for the backbone** of the Animation (ex. Tank) **with Google Drawings** for example.
2. **Add Polymer Web Components** to the SVG backbone **for visualization (and control?** Needs PLC access so too difficult**) of the process variables or PVs** (digital labels, gauges, slide bars, etc.). Data is retrieved in real-time from Databases (through secure OPC UA standards?). The **Logic to compare PVs with Set Points or SPs** can be performed **backend with either python or directly with html5:** “no script required! That's why it runs as a referenced image from this page. SVG graphics make it easy to generate vivid graphics with a minimal amount of download and trivial amounts of markup.”(<https://www.html5rocks.com/en/tutorials/svg/mobile_fundamentals/> ).
3. **Create Industrial System Unit: Custom Web Components = animable Backbone Module (SVG) + Process Data Visualization and Control Polymer Web Components. These Web**
4. **Test Units** (which are custom web components) **individually animated with mock-data.**
5. Create OPC UA compliant communication between **custom web components and databases.**
6. Repeat Steps until we have a complete **library of industrial module web components.**
7. Implement the **HTML5 Drag & Drop functionality** for the individual Modules (direct DOM manipulation through HTML5 easier for tracking events, alternately HTML5 Canvas.
8. PWA creation starting with PWA Polymer Starter Kit, modification of screens.
9. **Add sidebar with all industrial modules ordered** in categories according to process for easy finding).
10. PWA Testing with mock data
11. PWA Testing with real data
12. PWA User Account Management with Firebase?
13. PWA Deployment with Firebase?

**Why SVG? 🡪 Animate with ease:**

* One advantage of **SVG** is that the gradient itself **lives in the DOM**. This means that you can **modify it with script**, but more importantly you **can take advantage of SVG's built in animation capability to add subtle changes to your content.**
* **Embedding SVG (see object vs embed tag)** such that you get both **fallback to raster and ability to manipulate it by CSS** (https://stackoverflow.com/questions/25383527/embedding-svg-such-that-you-get-both-fallback-to-raster-and-ability-to-manipulat)

## Tools:

**Shadow DOM visualizer online** : <http://html5-demos.appspot.com/static/shadowdom-visualizer/index.html>

**Polymer:** The Polymer library provides a set of features for creating custom elements. These features are designed to make it easier and faster to make custom elements that work like standard DOM elements. Similar to standard DOM elements, Polymer elements can be.

* **Ich kenne mich damit schon gut aus.**
* HTML5 Web Components erlauben modulare und flexible **Erstellung von Bibliotheken von Komponenten die Industrielle Prozesse abbilden, modellieren, und als animierte Visualisierung darstellen. (WIE LEGO BLOCKS) (mit Ply)**
* Sichere Kommunikation mit OPC Systeme und Datenbanken möglich (**HyperUA**) oder via **Python back end.**
* Wiederverwendung der Web Components.
* Web Components **können individuell getestet werde**, dann zusammen in einem System, und dann in App.
* **Drag and Drop Functionality with Canvas or directly through DOM with HTML5**
* **OPC UA** (standard for industrial communications to allow information to be easily and securely exchanged between diverse platforms from multiple vendors)

# Ideen:

Modelorientierte/Objektorientierte anpassungsfähige **Automatisierung/MES/SCADA/HMI** fertigungstechnischer/verfahrenstechnischer Anlagen mit Python (und Microcontrollern) (zum **Ersetzen/Komplementieren** von SPS)

* **SCADA** (supervisory **control (of the PLC) and data adquisition**) with python (micropython) and microcontrollers (pyboard).
* **PLC replacement** with python (micropython) and microcontrollers (pyboard), with wireless connectivity to SCADA to send acquired data.
* **GUI part of SCADA (focus on** **adquisition and visualization** of data with less control of the process itself through PLC. So **basically PLC Indpendent**).
* **Python data communication protocol to extract data** from already functioning PLCs/SCADAs **(non invasive SCADA implementation to further expand functional Automated Systems and SCADAS.)**

# Themen die mich sehr interessieren:

* Automatisierung
  + Model-/Objektorientierte Automatisierung
* SCADA
  + UIs
    - GUI/HMI
  + Steuerung:
    - PLC
  + Daten
    - -erfassung
    - -verarbeitung
    - -visualizierung
* Augmented Reality
  + Web App
  + Predictive Maintainance
  + Process Flow Displays for workers
  + Path followers for Warehouse
  + Montageanimationen und Anleitungen (mit CAD Dateien)
  + Auftragerfassung
  + Datenvisualizierung (Plant Floor bzw. Leitebene)
* IoT / Industry 4.0
* Web Apps (Progressive Web Apps)
* Flexible, modular automation with microcontrollers
* Fertigungstechnik

# Tools die mich sehr interessieren:

* Python
* Microcontrollers (PyBoard and Micropython or other)
* Controllino (industry ready, freely programmable PLC)
* HTML5, JS, CSS (Web Components, Polymer)
* Industrial/standard communication buses and protocols

# Thesis Tittles:

* Industrial Web App for the flexible and modular creation of GUIs for industrial Manufacturing/Conveyor/Process lines with Integration to \_\_\_\_\_\_\_ Databases following the OPC UA
* Industrial Web App for the flexible and modular creation of GUIs for industrial Manufacturing/Conveyor/Process lines with Material Flow Compatibility.
* Leveraging HTML5 Web Components to create flexible Industrial Web Apps
* Library of Industrial Modules for flexible and modular deployment of GUIs through (Progressive) Web Apps and Web Components.

# Andere Themen:

* Python-opcua
* Python Factory Automation with PyBoard
* Embeddable SCADA for Factory Automation (view ESSA on github)
* Python databases
* Python as a PLC to control a process, record and send data, visualize data in a GUI…
* Python Industrial GUI either with PyQT or Django on the Web
* KPI dashboard web app Pyexcel to retrieve business data
* Python for process control, monitoring

# Reads:

* **PROJEXSYS**: Components and SOA for Industrial Web Apps