CLOSER 2018 Keynote: Lee Gillam, University of Surrey

Will Cloud Gain an Edge?

Or

CLOSER, to the Edge
‘Traditional’ Cloud

- NIST SP800-145 (aka Mell and Grance): 3-4-5 & SPI
- Large (economically efficient, easily maintained – but still expensive) datacentres in relatively few, geographical locations (regions) to support large user numbers, centralized corporate entity
- A ‘Big Four’ in Amazon, Microsoft, IBM, Google

‘New’ Cloud

- Containers (Docker, kernel-locked), and Functions (multiplicity of approaches) – added “CaaS” and “FaaS”
- **Edge** (multiplicity of approaches and concerns)
- (Re-)**distributed** Computing, and ‘new’ problems (new ‘traditional’ problems)
- ‘Big Four’?
Overview of this talk

‘Traditional’ Cloud
(Big Four) Clouds are big
Cost and performance (=cost) variation
Performance variation and implications for energy efficiency

‘New’ cloud
‘serverless’ and performance
Multiplicity of Edges
‘serverless’ Edges

An application
Cloud Cars and exemplars

Summary and take home
‘Traditional’ Cloud
(Illusion of) Infinite capacity - consider one (big) provider:

**AWS**: 2014 based on 11 regions and 28 AZs: 2.8-5.6m servers (Morgan, 2014) based on datacenter of up to 80k servers

12 Jun 2012, 1 trillion objects in S3.
13 April 2013, 2 trillion

2018, **18** regions, **54** AZs, 5 more regions coming: ∼**10m** servers?

Clouds are ‘big’

**Regions:** Contain big datacenters at distance but with big networking also

**Edges:** a shorter distance to something that does something useful for compute

Manchester United pitch ~80,000 sq ft

Figure sources: Datapath.io and Make IT Green: Cloud Computing and its Contribution to Climate Change, Greenpeace
## Clouds are ‘big’ – with cost variation

<table>
<thead>
<tr>
<th>Date</th>
<th>vCPU</th>
<th>ECU</th>
<th>Mem (GiB)</th>
<th>US-E (NV)</th>
<th>EU-W (Ire)</th>
<th>EU-W (Fra)</th>
<th>SA (SP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>27/4/16</td>
<td>t2.nano</td>
<td>1</td>
<td>Variable</td>
<td>0.5</td>
<td>$0.0065</td>
<td>$0.007</td>
<td>$0.0075</td>
</tr>
<tr>
<td></td>
<td>t2.micro</td>
<td>1</td>
<td>Variable</td>
<td>1</td>
<td>$0.013</td>
<td>$0.014</td>
<td>$0.015</td>
</tr>
<tr>
<td></td>
<td>m4.xlarge</td>
<td>4</td>
<td>13</td>
<td>16</td>
<td>$0.239</td>
<td>$0.264</td>
<td>$0.285</td>
</tr>
<tr>
<td></td>
<td>m4.2xlarge</td>
<td>8</td>
<td>26</td>
<td>32</td>
<td>$0.479</td>
<td>$0.528</td>
<td>$0.57</td>
</tr>
<tr>
<td></td>
<td>m4.4xlarge</td>
<td>16</td>
<td>53.5</td>
<td>64</td>
<td>$0.958</td>
<td>$1.056</td>
<td>$1.14</td>
</tr>
<tr>
<td></td>
<td>m4.10xlarge</td>
<td>40</td>
<td>124.5</td>
<td>160</td>
<td>$2.394</td>
<td>$2.641</td>
<td>$2.85</td>
</tr>
<tr>
<td></td>
<td>m3.medium</td>
<td>1</td>
<td>3</td>
<td>3.75</td>
<td>$0.067</td>
<td>$0.073</td>
<td>$0.079</td>
</tr>
<tr>
<td></td>
<td>m3.large</td>
<td>2</td>
<td>6.5</td>
<td>7.5</td>
<td>$0.133</td>
<td>$0.146</td>
<td>$0.158</td>
</tr>
<tr>
<td></td>
<td>m3.xlarge</td>
<td>4</td>
<td>13</td>
<td>15</td>
<td>$0.266</td>
<td>$0.293</td>
<td>$0.315</td>
</tr>
<tr>
<td></td>
<td>m3.2xlarge</td>
<td>8</td>
<td>26</td>
<td>30</td>
<td>$0.532</td>
<td>$0.585</td>
<td>$0.632</td>
</tr>
<tr>
<td></td>
<td>c4.large</td>
<td>2</td>
<td>8</td>
<td>3.75</td>
<td>$0.105</td>
<td>$0.119</td>
<td>$0.134</td>
</tr>
<tr>
<td></td>
<td>c4.xlarge</td>
<td>4</td>
<td>16</td>
<td>7.5</td>
<td>$0.209</td>
<td>$0.238</td>
<td>$0.267</td>
</tr>
<tr>
<td></td>
<td>c4.2xlarge</td>
<td>8</td>
<td>31</td>
<td>15</td>
<td>$0.419</td>
<td>$0.477</td>
<td>$0.534</td>
</tr>
<tr>
<td></td>
<td>c4.4xlarge</td>
<td>16</td>
<td>62</td>
<td>30</td>
<td>$0.838</td>
<td>$0.953</td>
<td>$1.069</td>
</tr>
<tr>
<td></td>
<td>c4.8xlarge</td>
<td>36</td>
<td>132</td>
<td>60</td>
<td>$1.675</td>
<td>$1.906</td>
<td>$2.138</td>
</tr>
<tr>
<td></td>
<td>c3.large</td>
<td>2</td>
<td>7</td>
<td>3.75</td>
<td>$0.105</td>
<td>$0.12</td>
<td>$0.129</td>
</tr>
<tr>
<td></td>
<td>c3.xlarge</td>
<td>4</td>
<td>14</td>
<td>7.5</td>
<td>$0.21</td>
<td>$0.239</td>
<td>$0.258</td>
</tr>
<tr>
<td></td>
<td>c3.2xlarge</td>
<td>8</td>
<td>28</td>
<td>15</td>
<td>$0.42</td>
<td>$0.478</td>
<td>$0.516</td>
</tr>
<tr>
<td></td>
<td>c3.4xlarge</td>
<td>16</td>
<td>55</td>
<td>30</td>
<td>$0.84</td>
<td>$0.956</td>
<td>$1.032</td>
</tr>
<tr>
<td></td>
<td>`c3.8xlarge</td>
<td>32</td>
<td>108</td>
<td>60</td>
<td>$1.68</td>
<td>$1.912</td>
<td>$2.064</td>
</tr>
</tbody>
</table>
Clouds are ‘big’ – with hardware variation

**CPU model** discovery - for ~700 EC2 FGS Instances for 1 user – **1 instance type**

<table>
<thead>
<tr>
<th>Region</th>
<th>AZ</th>
<th>E5430</th>
<th>E5-2650</th>
<th>E5645</th>
<th>E5507</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>US East N.Virginia 2006</strong></td>
<td><em>us-east-1a</em></td>
<td>31%</td>
<td>0</td>
<td>25%</td>
<td><strong>44%</strong></td>
</tr>
<tr>
<td>[year Region started]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cheapest – but latencies</td>
<td><em>us-east-1b</em></td>
<td>5%</td>
<td><strong>59%</strong></td>
<td>29%</td>
<td>7%</td>
</tr>
<tr>
<td></td>
<td><em>us-east-1c</em></td>
<td>0</td>
<td>47%</td>
<td><strong>52%</strong></td>
<td>1%</td>
</tr>
<tr>
<td></td>
<td><em>us-east-1d</em></td>
<td>18%</td>
<td>31%</td>
<td><strong>44%</strong></td>
<td>7%</td>
</tr>
<tr>
<td><strong>EU West Dublin 2007</strong></td>
<td><em>eu-west-1a</em></td>
<td>4%</td>
<td><strong>75%</strong></td>
<td>19%</td>
<td>2%</td>
</tr>
<tr>
<td></td>
<td><em>eu-west-1b</em></td>
<td>28%</td>
<td>0</td>
<td><strong>44%</strong></td>
<td>28%</td>
</tr>
<tr>
<td></td>
<td><em>eu-west-1c</em></td>
<td>4%</td>
<td>0</td>
<td><strong>63%</strong></td>
<td>33%</td>
</tr>
<tr>
<td><strong>US West N. California 2009</strong></td>
<td><em>us-west-1b</em></td>
<td>0</td>
<td>0</td>
<td>13%</td>
<td><strong>87%</strong></td>
</tr>
<tr>
<td></td>
<td><em>us-west-1c</em></td>
<td>8%</td>
<td>0</td>
<td>18%</td>
<td><strong>74%</strong></td>
</tr>
<tr>
<td><strong>SA San Paulo 2011</strong></td>
<td><em>sa-east-1a</em></td>
<td>0</td>
<td><strong>81%</strong></td>
<td>19%</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td><em>sa-east-1b</em></td>
<td>0</td>
<td><strong>86%</strong></td>
<td>14%</td>
<td>0</td>
</tr>
<tr>
<td><strong>US West Oregon 2011</strong></td>
<td><em>us-west-2b</em></td>
<td>0</td>
<td><strong>73%</strong></td>
<td>27%</td>
<td>0</td>
</tr>
<tr>
<td><strong>Asia Pacific Sydney 2012</strong></td>
<td><em>ap-southeast-2a</em></td>
<td>0</td>
<td><strong>64%</strong></td>
<td>36%</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td><em>ap-southeast-2b</em></td>
<td>0</td>
<td><strong>75%</strong></td>
<td>25%</td>
<td>0</td>
</tr>
</tbody>
</table>

Cost, latency, computational capability (moving up stack)
Performance varies by CPU

- **int** – left better
- **STREAM** – right better
- **fp** – left better
(at minimum) a user needs to:

- Identify suitable (cost-based?) instance offerings *(price determination)*
- Rank ‘best’ by workload *(performance determination)*
- Determine AZs (latency) offering those resources *(location selection)*
- Attempt to obtain them *(instance lottery)*
- For users with more than one account this may need to be done per account basis *(account selection)*

- **Costs** are incurred in *(1) performance determination* *(2) location selection* and *(3) instance lottery*, for every *(4) account selection* *(AND infrastructure composition changes over time)*

**Spend involved with getting cost-efficiency (performance gaming / deploy-and-ditch) - potentially expensive!**
Can’t eliminate resource uncertainty

Performance can be ‘stable’ over a long period for a given benchmark – past a good indicator of future - but may be subject to abrupt changes and severe degradation

One instance, **1379s** for POV-Ray – ~13 standard deviations from the mean (639, 54)

Rarer: ‘The requested Availability Zone is currently constrained and we are no longer accepting new customer requests for X/Y/Z instance types’

• Go elsewhere, but other AZs may not be cost-efficient – **AZ lock-in**

Unusually for a service: better can be cheaper

But much work around performance continues to assume homogeneity

Cost-efficient use
Cloud Service Brokers (e.g. aggregators) might address performance issues

**Instances with known performance characteristics**
- A match-making service between user application performance needs and available resources

**Re-price based on desired performance**
- Make more suitable instances more expensive, and less suitable less so.

**Extensive simulations suggested:**
- Assuming clouds are opaque makes it difficult to avoid instance gaming.
- **Very difficult to make a profit**, even with careful pool management! - high vol.
- Opportunities in value of utility rather than price

**Rare to find discussion of broker profit – Rogers & Cliff has been a notable exception**

Performance and energy trade-off for different kinds of workload

- *runtime variable with hardware (heterogeneity)*
- how much power needed to deliver runtime on given hardware
- best performance might not equate to most energy efficient
- performance $\rightarrow$ runtime; user cost higher with longer runtime

Put workloads on best machines for it: consolidation (implies migration)

- *risk of being own noisy neighbour* $\rightarrow$ longer runtimes
- additional energy use for period of migration: at least costs of equivalent resources plus network
- question of recoverability depends on continued use
- for providers, opportunity to switch off / maintain (may not be an incentive to)
Workload-related CPU model ranking
- w1: A > B > C > D;  w2: D > C > B > A
- VM allocation: B allocated as available; A preferred
- VM consolidation: migration beneficial if workload can recoup cost of migration – implies performance maintained (contention)

Consolidation with Migration Cost Recovery (CMCR)
- Migrate to more efficient hosts
- VM terminated before $[t_{off}]$, the effort is wasted
- Recover migration overhead $[\text{Cost}_{\text{mig}}]$, efficient gain after $[t_{off}]$
Broad characterisation: 9 scheduling approaches, several types of consolidation (+ none), CloudSim model using 12,583 heterogeneous hosts, 25m VMs (tasks) in Google workload trace data, 5 minimum-runtime settings, migration rounds at 5 minute intervals (host utilization < 20%). **On-demand** VM allocation

CPU info not provided, so map Google priorities to benchmark results (range scaled) for preferences (Gratis (0) : POVRAY, Batch (2) : NAMD, Production (9) : STREAM). *Distributions typically skewed lognormal per CPU model.*

Then relate power ratings of these ‘Cloud’ CPUs – simulation results can be related to real VMs.

<table>
<thead>
<tr>
<th>Workload</th>
<th>Benchmark</th>
<th>CPU Model</th>
<th>(μ)</th>
<th>(σ)</th>
<th>Min</th>
<th>Max</th>
<th>CoV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gratis</td>
<td>POVRAY</td>
<td>E5-430</td>
<td>439</td>
<td>11</td>
<td>421</td>
<td>467</td>
<td>0.025</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E5-2650</td>
<td>468</td>
<td>12</td>
<td>451</td>
<td>500</td>
<td>0.026</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E5-645</td>
<td>507</td>
<td>10</td>
<td>490</td>
<td>535</td>
<td>0.02</td>
</tr>
<tr>
<td>Batch</td>
<td>NAMD</td>
<td>E5-2651</td>
<td>1994</td>
<td>41.9</td>
<td>1952</td>
<td>2036</td>
<td>0.021</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E5-2650</td>
<td>2007</td>
<td>28.5</td>
<td>1978</td>
<td>2036</td>
<td>0.014</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E5-645</td>
<td>2043</td>
<td>96.4</td>
<td>1946</td>
<td>2140</td>
<td>0.047</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E5-430</td>
<td>2160</td>
<td>20.7</td>
<td>2135</td>
<td>2189</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E5-507</td>
<td>2187</td>
<td>18.1</td>
<td>2162</td>
<td>2217</td>
<td>0.008</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E5-645</td>
<td>3395</td>
<td>287</td>
<td>2995</td>
<td>4008</td>
<td>0.085</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E5-2650</td>
<td>5294</td>
<td>191</td>
<td>4935</td>
<td>5860</td>
<td>0.036</td>
</tr>
<tr>
<td>Production</td>
<td>STREAM</td>
<td>E5-430</td>
<td>1446</td>
<td>66</td>
<td>1328</td>
<td>1572</td>
<td>0.045</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E5-507</td>
<td>2348</td>
<td>104</td>
<td>2078</td>
<td>2448</td>
<td>0.044</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E5-645</td>
<td>3395</td>
<td>287</td>
<td>2995</td>
<td>4008</td>
<td>0.085</td>
</tr>
<tr>
<td></td>
<td></td>
<td>E5-2650</td>
<td>5294</td>
<td>191</td>
<td>4935</td>
<td>5860</td>
<td>0.036</td>
</tr>
</tbody>
</table>

**Google data**

<table>
<thead>
<tr>
<th>(μ)</th>
<th>(σ)</th>
<th>Min</th>
<th>Max</th>
<th>CoV</th>
</tr>
</thead>
<tbody>
<tr>
<td>438.06</td>
<td>9.42</td>
<td>421</td>
<td>467</td>
<td>0.022</td>
</tr>
<tr>
<td>473.87</td>
<td>11.93</td>
<td>451</td>
<td>500</td>
<td>0.026</td>
</tr>
<tr>
<td>498.55</td>
<td>10.44</td>
<td>490</td>
<td>535</td>
<td>0.021</td>
</tr>
<tr>
<td>1991</td>
<td>39.51</td>
<td>1800</td>
<td>2040</td>
<td>0.02</td>
</tr>
<tr>
<td>1963.4</td>
<td>28.41</td>
<td>1900</td>
<td>2015</td>
<td>0.015</td>
</tr>
<tr>
<td>1931.4</td>
<td>93.43</td>
<td>1800</td>
<td>2170</td>
<td>0.048</td>
</tr>
<tr>
<td>2103.6</td>
<td>22.1</td>
<td>2080</td>
<td>2190</td>
<td>0.011</td>
</tr>
<tr>
<td>2191.8</td>
<td>15.69</td>
<td>2150</td>
<td>2200</td>
<td>0.007</td>
</tr>
<tr>
<td>1404.4</td>
<td>43.33</td>
<td>1328</td>
<td>1572</td>
<td>0.032</td>
</tr>
<tr>
<td>2346.7</td>
<td>107.21</td>
<td>2078</td>
<td>2448</td>
<td>0.046</td>
</tr>
<tr>
<td>3388.7</td>
<td>238.22</td>
<td>2995</td>
<td>4008</td>
<td>0.07</td>
</tr>
<tr>
<td>5294.5</td>
<td>197.52</td>
<td>4935</td>
<td>5860</td>
<td>0.037</td>
</tr>
</tbody>
</table>
Findings confirmed: *sensible* allocation better (easier) than consolidation; migrate longer running VMs – but **assumes clouds are not opaque**. (i.e. provider has knowledge of workload)

‘New’ Cloud
Relevance to ‘new’ Cloud

A look at so-called ‘Function as a Service’

‘serverless’ computing (yet servers are **essential**)

– You’re not supposed to “worry” about provisioning
– Billing per 100 milliseconds (AWS Lambda, Google Cloud Functions; Azure has at least 2 ways to pay incl. based on memory consumption)
– Functions may be time-constrained – 5 mins, Lambda/Azure; though HTTP timeout of 30s (e.g. AWS API Gateway) gives 29s runtime

**Are performance/cost questions relevant?**
A look at so-called ‘Function as a Service’ – AWS Lambda

AWS Lambda runs a Function in a Container on a VM (‘serverless’)

IP address may over time change – 2 functions run gave e.g.: ip-10-23-17-3, ip-10-14-98-122.

Short runtimes good: a small test - 113.44 ms, then 114.34 ms, 102.65 ms, 113.57 ms – all rounded to nearest 100 (200ms) for billing; reasonable consistency.

Underlying process:

<table>
<thead>
<tr>
<th>USER</th>
<th>PID</th>
<th>%CPU</th>
<th>%MEM</th>
<th>VSZ</th>
<th>RSS</th>
<th>TTY</th>
<th>STAT</th>
<th>START</th>
<th>TIME</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>490</td>
<td>1</td>
<td>1.3</td>
<td>0.3</td>
<td>212024</td>
<td>15372</td>
<td>?</td>
<td>Ss</td>
<td>16:49</td>
<td>0:00</td>
<td>/usr/bin/python2.7 /var/runtime/awslambda/bootstrap.py</td>
</tr>
<tr>
<td>490</td>
<td>7</td>
<td>0.0</td>
<td>0.0</td>
<td>117208</td>
<td>2476</td>
<td>?</td>
<td>R</td>
<td>16:49</td>
<td>0:00</td>
<td>ps auxw</td>
</tr>
</tbody>
</table>

*Through several uses, underlying process remains.*
Limitations exist, e.g. can’t run ‘ifconfig’, no ‘sudo’ so can’t install as root, and can’t get at AWS metadata of VM. But can find out CPU model (/proc/cpuinfo [dual ‘core’ c4]) and underlying system (uname [AWS Linux]),

model name : Intel(R) Xeon(R) CPU E5-2666 v3 @ 2.90GHz 
cpu MHz : 2900.066 
cache size : 25600 KB

Linux ip-10-23-17-3 4.4.35-33.55.amzn1.x86_64 #1 SMP ... 
x86_64 x86_64 x86_64 GNU/Linux

Others have seen:

– Intel(R) Xeon(R) CPU E5-2680 v2 @ 2.80GHz - a c3 instance: http://zqsmm.qiniucdn.com/data/20150416152509/index.html.
Can run a [small –time limit!] arbitrary linux application – e.g. a benchmark such as STREAM [\(~2\text{GB/s}\)], if:

- Precompiled elsewhere, downloaded into local filestore for Function (/tmp), made executable (chmod), executed and output returned
- Variations \textit{per execution}, with rounding; \textit{performance/location/lottery/account remains important}

STREAM version $\text{Revision: 5.10}$
**Multiplicity of Edges**

Will Cloud gain an **Edge**?

Network **Edge devices** – devices that have sensors (a mobile phone?)

Customer **Edge** (/Edge router) – router on premises

Provider **Edge** – a provider’s router

**Edge** Datacenter (/**Cloudlet***/Content Delivery Network) – datacenter

Multi-access (/mobile) **Edge** – datacenter + RAN (e.g. 5G)

Also, **FOG**

*Edge is also Microsoft’s web browser, and a Ford vehicle*

*Should not be confused with Cloudlet in CloudSim, which is a Task*
Fog “complements and extends the Cloud to the edge and endpoints”, Bonomi et al

- Fog is additional to and complementing Cloud,
- Example distributed applications: “A has one Cloud component, and two Fog components [...] B has one cloud component, one component in the Core, and a Fog component”.

OpenFog Consortium: “a system-level horizontal architecture that distributes resources and services of computing, storage, control and networking anywhere along the continuum from Cloud to Things”.

- OpenFog documents represent Fog diagrammatically as:
  - between Cloud and Things;
  - including Cloud;
- OpenFog Consortium: Fog is “often erroneously called edge computing, but there are key differences. Fog works with the cloud, whereas edge is defined by the exclusion of cloud”.

Fog has not proven an entirely helpful notion
Multi-access Edge Computing (until recently, Mobile Edge Computing) – ETSI specification

- provide **capabilities of Cloud Computing close to the Radio Access Networks** in 4G and 5G telecommunications and converge with other radio access technologies (e.g. WiFi or Satellite).
- “can be seen as a **cloud server running at the edge of a mobile network**”.

ETSI MEC server supports VMs into which “MEC applications from vendors, service providers and third-parties are deployed and executed”.

*VMs → Containers → Functions [MEC authors: PaaS for “future releases”]*.
Not just one MEC server per Edge?

**Cloudlet** (Carnegie Mellon University (CMU) notion) - middle tier of a 3-tier hierarchy: “mobile device - cloudlet – cloud”.

A “**datacentre in a box**”, implying multiple servers, local to the user, similar to MEC (and some characterisations of Fog).

CMU formed Open Edge Computing initiative in 2015, together with Intel, Huawei, and Vodafone, intending to synchronize work with ETSI MEC, leading so far to OpenStack++ allowing for migration between OpenStack clusters.

*Integration with telecommunications per MEC does not appear yet to be paralleled.*
Intended benefits of Edge: reduced end-to-end latency; smaller data volumes travelling shorter distances – computational capability and storage is nearer the user.

Intended benefits of Functions: small, fast-executing, provider-scaled capability.

Faster, smaller implies more suitable for cloud-assisted, or cloud-driven, control services.

But: susceptible to hardware variations (performance), including due to contention, as well as provider-driven energy management.

See, also, AWS Lambda at Edge (CDN-related FaaS).
Edges running VMs / Containers / Functions, using various data

**Migration** when user moves to another edge

Execution would vary with hardware (slowdown/speedup may imply needing more/less resource for equivalence at the target – and need to know this)

Is a user highly likely to keep moving quickly between edges?
An Application
Connected and Autonomous Vehicles

Socio-economic forces

Safety
• 8m accidents, 1.3m fatalities, 7m injuries*

Economy
• 90b hours in traffic jams*

Environment
• 20-35% of global CO2 emission

Market forces

*CISCO report
(How) Can cars become fully independent?

What technology advances are required?

How will humans and vehicles interact with each other and their environment?
Jaguar Land Rover and EPSRC announce £11 million autonomous vehicle research programme

Related links
- RAS 2020 Robotics and Autonomous Systems (PDF)
- Jaguar Land Rover
- University of Surrey
- Waroc
- Jaguar
- Cranfield University
- Southamp
Internet of Things (Tractors, Kettles, Fridges, Cars ....)

Waymo generates 1GB data per second (2 PB/car/year)

Estimate of 2 billion total vehicles by 2020

Some of this 1GB/s may be usefully processed close to the vehicles, with vehicles connecting through the RAN to access or provide e.g. local road information

Some data may aggregated over MEC Servers, or where capability is not needed immediately, or if not available locally
CARMA’s vision: design and validate a novel, secure framework to enable implementation of safe and robust semi-autonomous and fully autonomous functions

**Main objectives**
- Address key technical research challenges
- Validate through proof-of-concept demonstrators
- Evaluate scalability

**Multi-access/mobile edge, 5G, Cloud.**
*Security and effects on performance and latency*

*Safety remains paramount*
CARMA Core (Cloud):
• Based on commercially available public cloud resources
• Services where higher latency is tolerable, information is coming from a wider geography, longer term storage needs, and so on.

CARMA Edge (MEC, 5G):
• Host beneficially off-board (low latency) processes
• Information collected from around vehicles to support cooperativity and computation beyond capabilities, including sensor ranges, of a given vehicle.

CARMA Vehicle:
• On-board vehicle network across all sensors, infotainment, actuators etc.
• Significant increases in on-board computational capability to be expected.
Cooperativity beneficial for maps where information has varying transience – information around vehicles and beyond sensor ranges
CARMA Platform: Logical Design
IBM Cloud-connected Vehicles

Largely Vehicle to Cloud
- IBM Watson IoT Driver Behavior Service
- IBM Connected-car IoT app with Geospatial Analytics.
- Microsoft Connected Vehicle Platform
- Google Connected Vehicle Platform
- ....

A hint of Edge, in -
- AWS Connected Vehicle Solution
- IBM Edge (Apache Edgent) analytics 
  ...

None of these address Autonomous
AWS suggests an application stack launchable from a template for multiple services
- Long term data storage
- Treatment of telematics anomalies
- Capture of trip data
- Calculation of a driver safety score
- Diagnostics and reporting

Messages through IoT gateway (Message Queue Telemetry Transport – MQTT, pub/sub): hierarchy of topics, and data payloads; rules on messages trigger execution of Lambda Functions for applications.

Includes AWS Greengrass for the **Edge**, but **most of the work in Cloud**
Public Cloud Exemplars

raw data
Amazon S3

anomaly detection
Amazon Kinesis Analytics
Amazon Kinesis Streams
AWS Lambda
Amazon DynamoDB

trip data
AWS Lambda
Amazon DynamoDB

driver safety score
AWS Lambda
Amazon SNS

diagnostic trouble codes
AWS Lambda
Amazon DynamoDB

just-in-time registration
AWS Lambda

connected vehicles with AWS Greengrass Core

Location-based marketing

notification service

University of Surrey
IBM Edge Analytics: Edge Agent on a Raspberry Pi and the DGLux tool.

How Edge Analytics works
Edge offerings largely not yet replicating Clouds; actions, per provider, required in order to achieve.

- **AWS** IoT rules not yet deployable to (Greengrass) Edge; local versions of other services largely also unavailable.
- **IBM** “Edge Analytic Rule(s) are pushed to the edge device” but no other services
- **Microsoft IoT Edge** requires a device running Windows 10 or Windows server with Docker
- **Google IoT** seems not to relate to Edge capability as yet.
- Vendors also promoting their own flavour of Function as a Service.

MQTT (Advanced Message Queuing Protocol), tending to be supported – AMQP and others less so.

This is an improving situation.
Summary and take home
Summary

Will Cloud gain an Edge?

Which Edge? For us, MEC, but market forces…

`Traditional’ Cloud
(Big Four) Clouds are big
Cost and performance (=cost) variation
Performance variation and implications for energy efficiency

`New’ cloud
’serverless’ and performance
Multiplicity of Edges
’serverless’ Edges

An application
Cloud Cars and exemplars

Summary and take home
Provider challenges include:

– **What to provide in hardware**
  - light lifting (rPis)
  - heavy lifting (servers, stacks)
  - telecommunications connectivity
  - “Moore’s law”

– **What to provide on hardware**
  - Bare metal
  - VMs/Containers/Functions

– Where to locate
– How to maintain
– Support for **Migration**
– How to secure
– To what quality of service
– At what price
Application/user challenges, as well as services to prioritise, include:

– Which Clouds/Edges to use? Vendor(s) lock-in? (hostage to functionality)
– What runs where in V/E/C, when?
  • Fixed, or dynamic, accounting for limited, heterogeneous, resources under contention? (dynamic reconfiguration)
  • What Performance – guarantees?
– Migration between V/E/C and/or across Edges? (c/w elastic / scalable)
  • Live migration – uninterruptible?
– How to secure? (What to secure? When to secure?)
– How to price across V/E/C? Who pays whom for what?
  • Fixed / dynamic
  • Based on services and/or demand?
An opportunity for an interested researcher ....

**Cloud Assisted Real-time Methods for Autonomy (CARMA)**

Research Fellow in Mobile/Multi-access Edge Cloud Computing

**Deadline: 8 April**

https://jobs.surrey.ac.uk/Vacancy.aspx?id=4701
• P. Mell, T. Grance, NIST Definition of Cloud Computing, NIST, 2011
Selected publications

- John O'Loughlin and Lee Gillam (2017) "A Performance Brokerage for Heterogeneous Clouds". FGCS.
Thank You

l.gillam@surrey.ac.uk

Further information:

Publications: https://sites.google.com/site/drleegillam/publica
Twitter: @leegillam
Journal: http://www.journalofcloudcomputing.com/ - fully open access to in-depth Cloud research

https://jobs.surrey.ac.uk/Vacancy.aspx?id=4701

Acknowledgements:
Drs. O’Loughlin & Zakarya; CARMA team, JLR, EPSRC
And, where needed, Google Images