PRINCETON UNIVERSITY

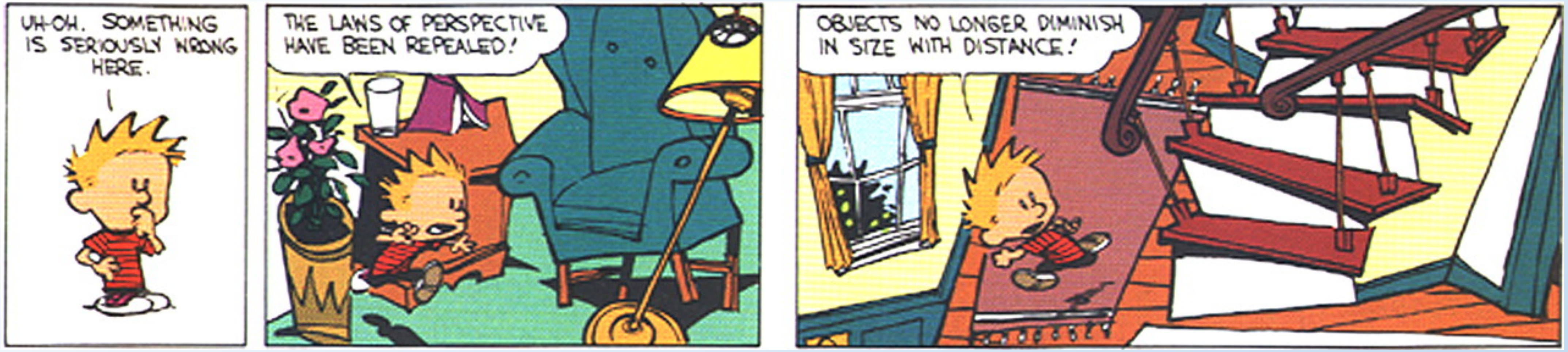
CMS Computing Resources: Meeting the demands of the high-luminosity LHC physics program

Ken Bloom, Tommaso Boccali, Oliver Gutsche,
David Lange, Eric Vaandering

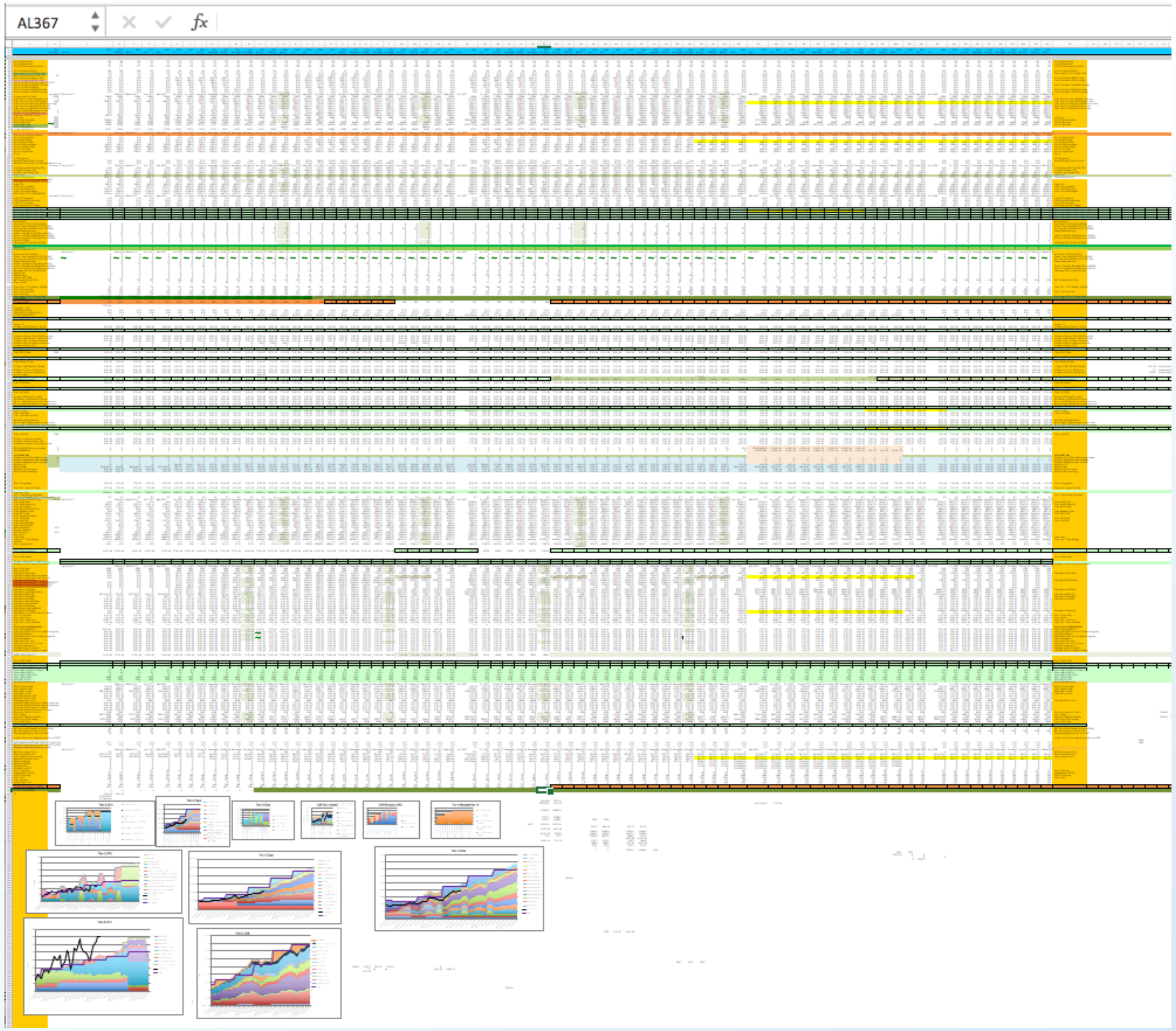
July 12, 2018

Why do we need tools for projecting computing resource needs?

- Provide input to external reviews
  - Short-term estimates of needs (Typically CPU, disk, tape by computing tier)
  - Long-term projections for Run 3 and HL-LHC
- Understand the implications of evolutions to computing models
  - Identify critical components that drive resource needs
  - **Demonstrate impact of physics choices and R&D activities**



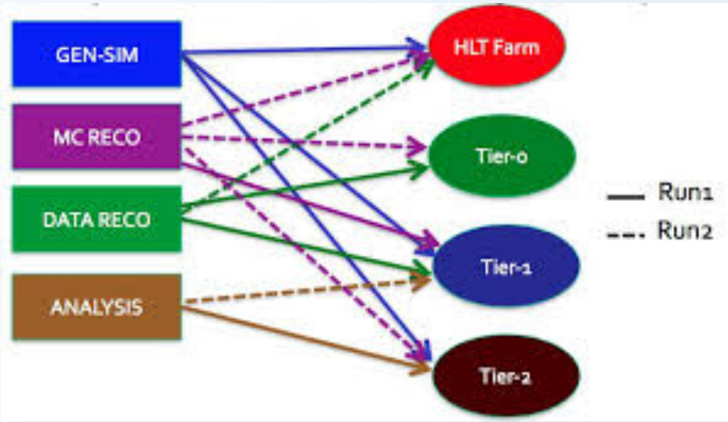
Spreadsheet models developed by LHC experiments for computing resource need estimates are very complex



- Traditionally: Detailed projections of resource needs with 2-3 year horizon are reviewed every 6 months
- Initially simple models have become complex to meet growing demands for accuracy, fidelity and timescales

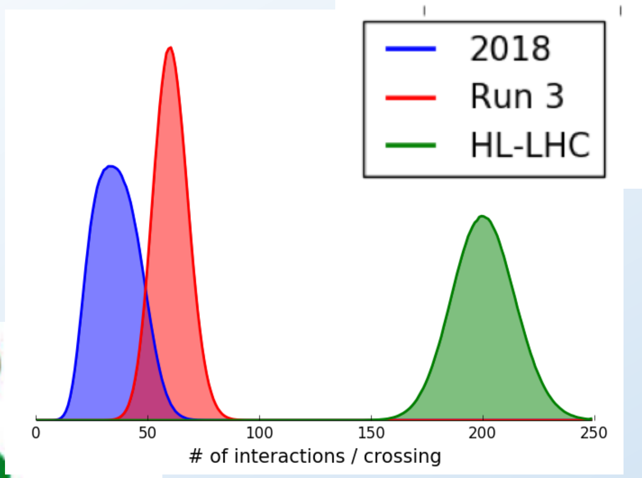
This complexity comes from having a truly complex environment and the need to model it in detail

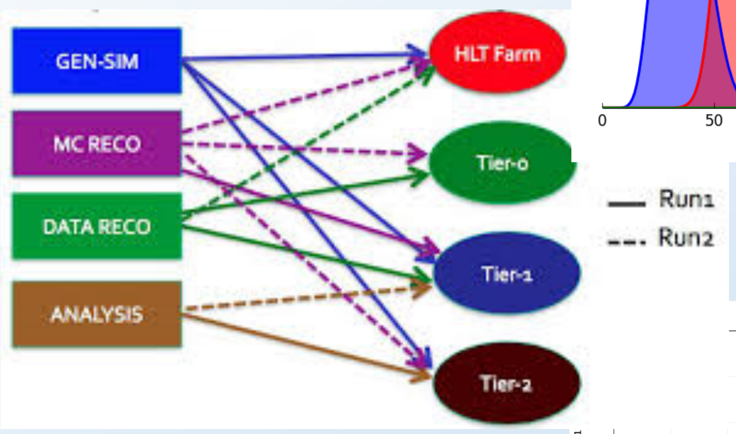
- What workflows to run? Where? and when?
- Evolution of experiment workflows, data tiers, analysis requirements, etc

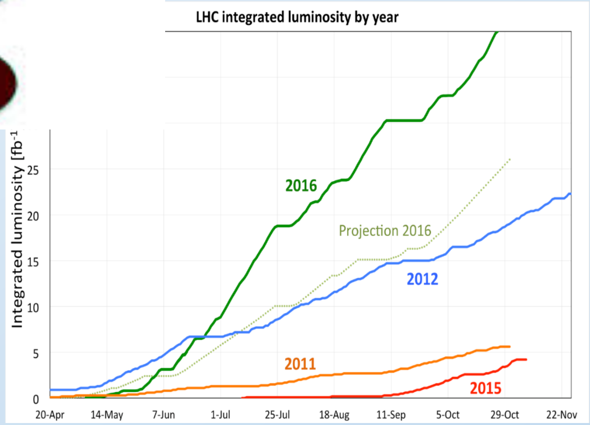


This complexity comes from having a truly complex environment and the need to model it in detail

- What workflows to run? Where? and when?
- Evolution of experiment workflows, data tiers, analysis requirements, etc
- Evolution with instantaneous luminosity
- Evolution with integrated luminosity
- Impact of LHC reliability

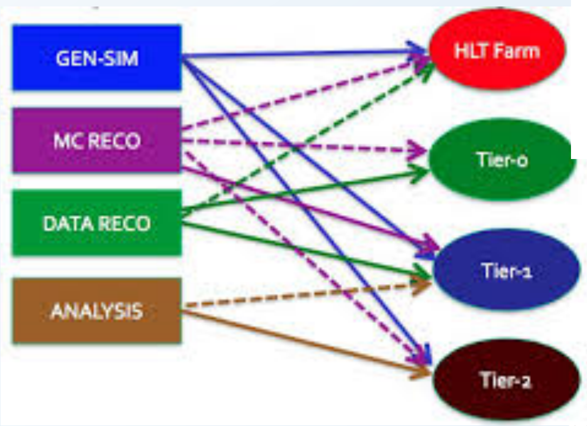


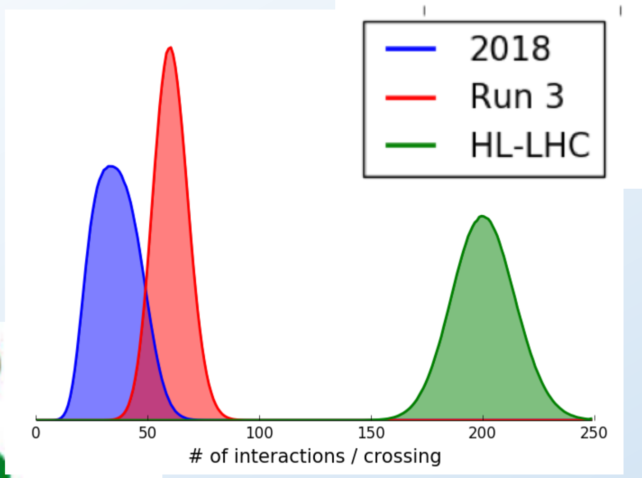


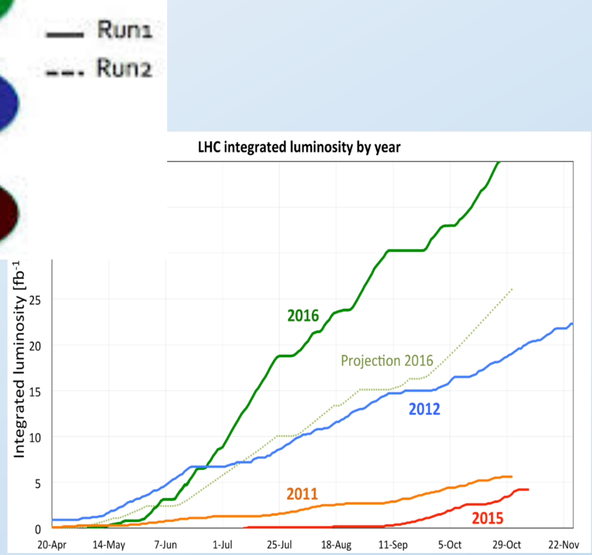


This complexity comes from having a truly complex environment and the need to model it in detail

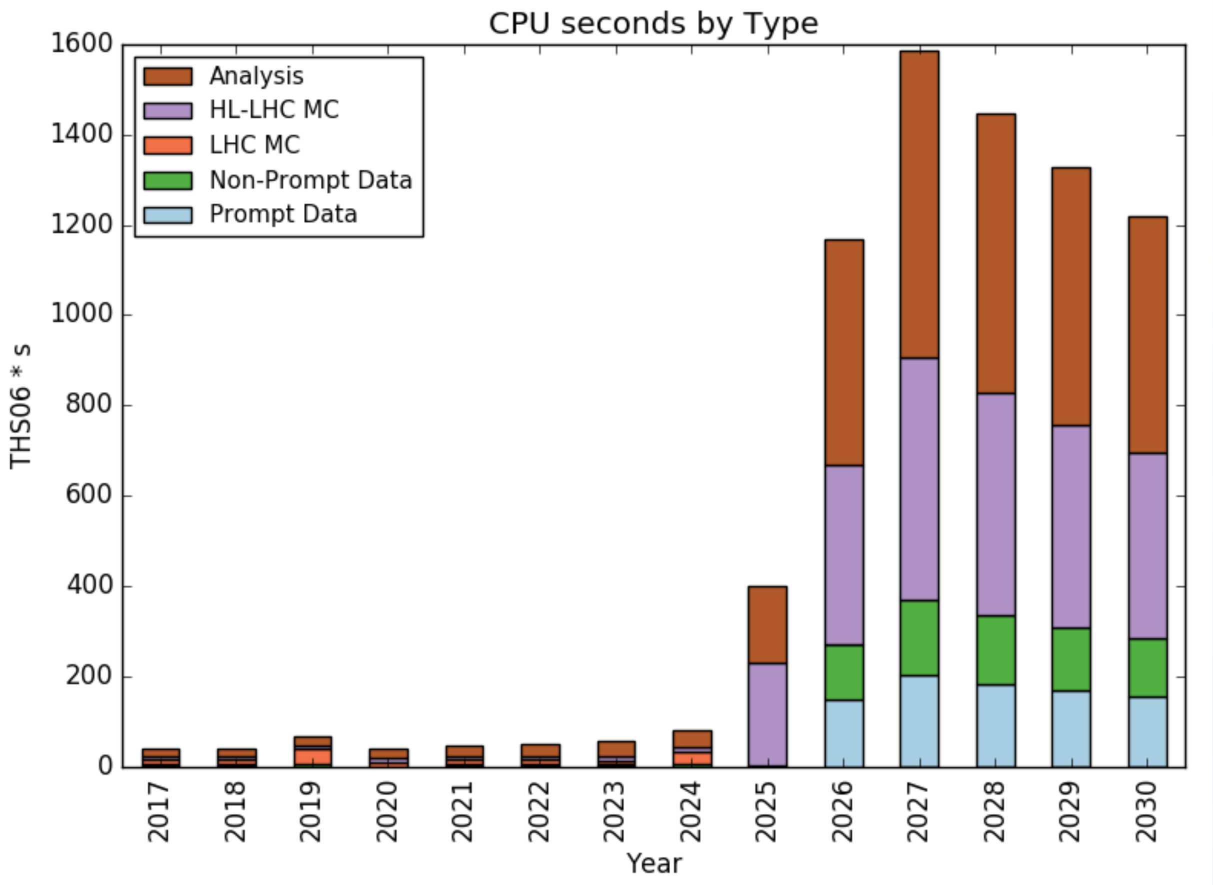
- What workflows to run? Where? and when?
- Evolution of experiment workflows, data tiers, analysis requirements, etc
- Evolution with instantaneous luminosity
- Evolution with integrated luminosity
- Impact of LHC reliability
- Expected analysis user behavior
- Evolving balance of commissioning needs vs analysis needs
- Impact of site infrastructure needs
- Use of dynamic and heterogeneous resources
- Policies that ensure efficient resource usage (e.g., data management policies)

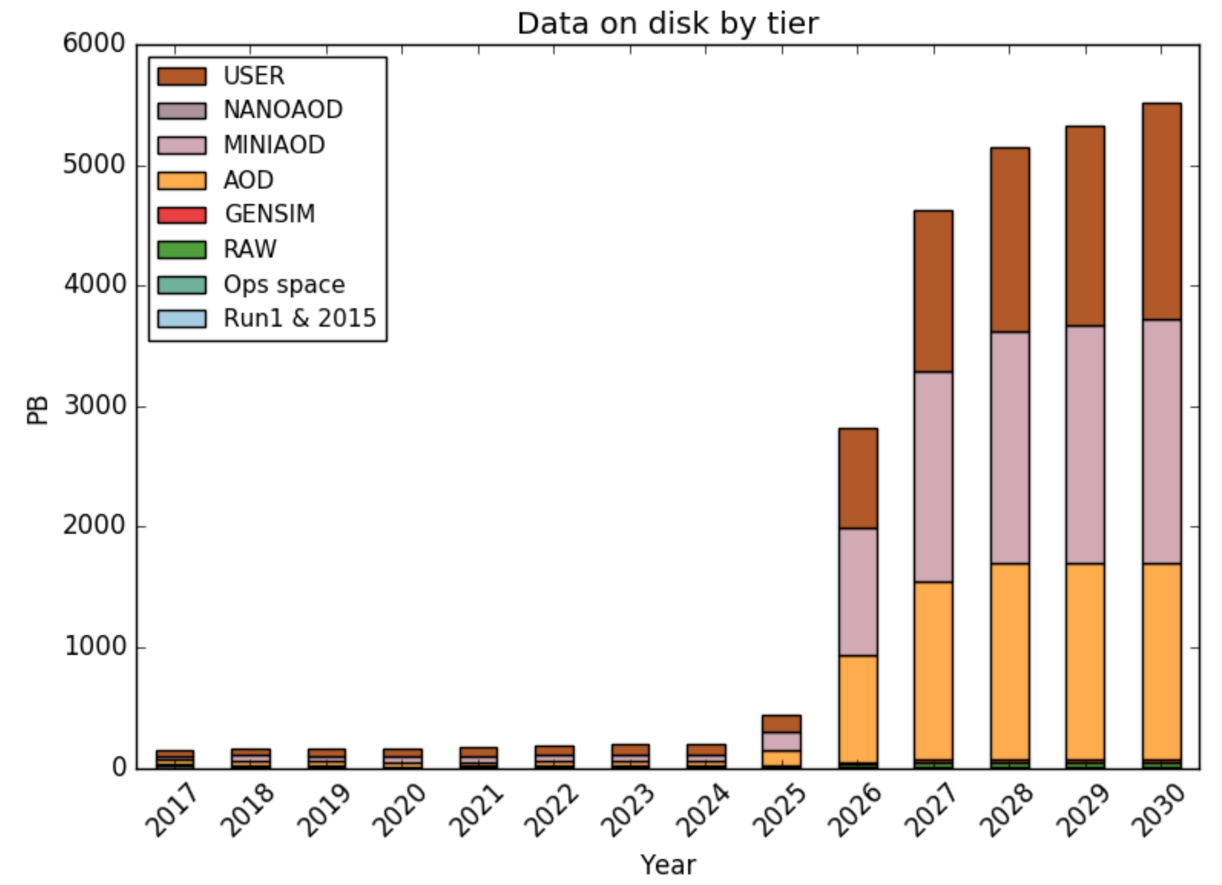






Long term (and thus naïve) projections are at lower fidelity





The idea behind models is quite simple

All the work we want to do

Map resources to activities

$$Resource_i\ (t) = \sum_{j}^{Activities} Resource_{ij}\,[t, LHC(t), HLT(t), PM(t), AM(t), TE(t), policies, \ldots]$$

Total need for resource type i

The idea behind models is quite simple

All the work we want to do

Map resources to activities

Trigger rates into offline

Analysis model

$$Resource_i\ (t) = \sum_{j}^{Activities} Resource_{ij}\,[t, LHC(t), HLT(t), PM(t), AM(t), TE(t), policies, \ldots]$$

Total need for resource type i

Machine plans + performance

Production model

Technology evolution

The idea behind models is quite simple

All the work we want to do

Map resources to activities

Trigger rates into offline

Analysis model

$$Resource_i\ (t) = \sum_{j}^{Activities} Resource_{ij}\,[t, LHC(t), HLT(t), PM(t), AM(t), TE(t), policies, \ldots]$$

Total need for resource type i

Machine plans + performance

Production model

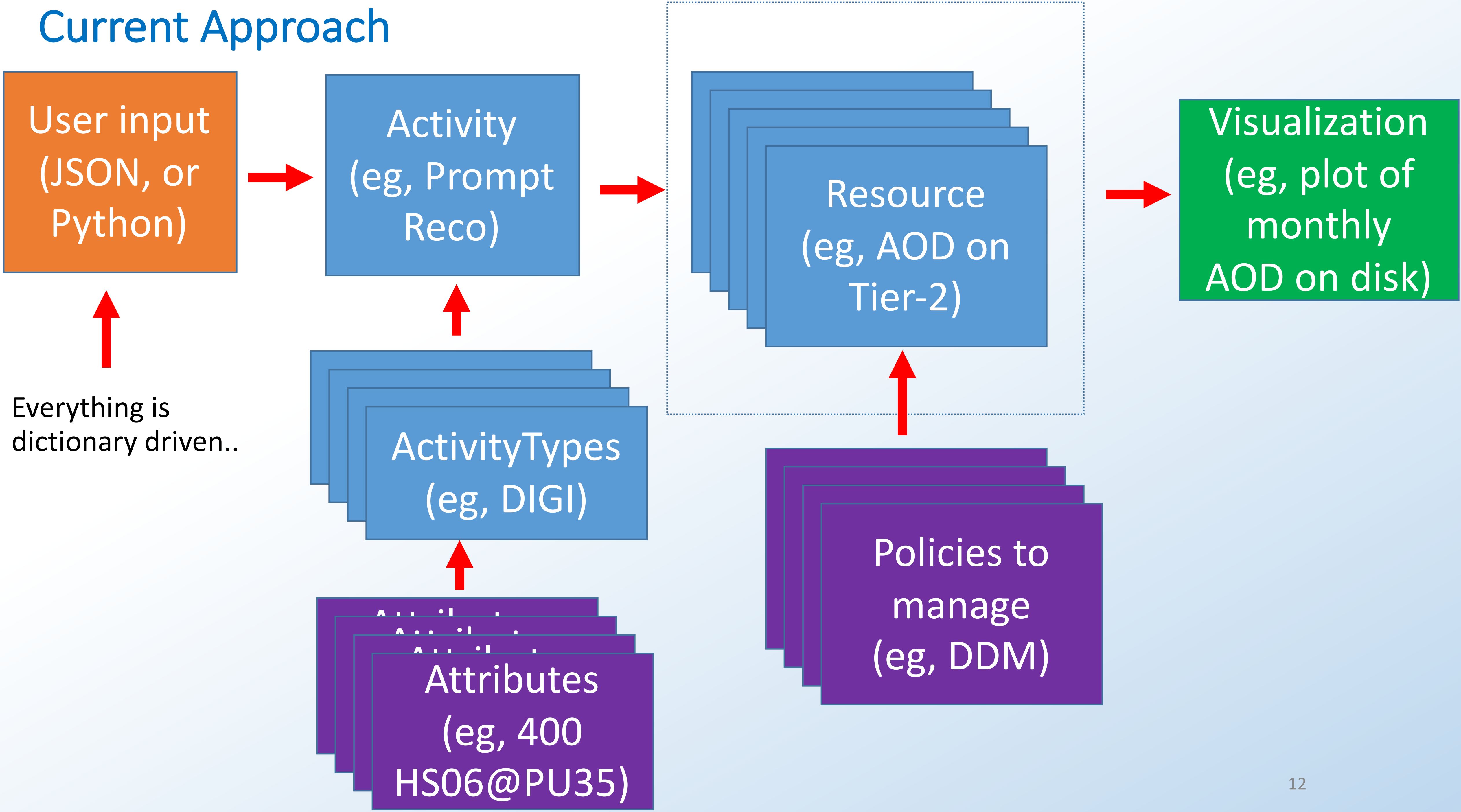
Technology evolution

Resource management

New Implementations are needed as we increase the fidelity of Run 3 and HL-LHC models

- CMS has chosen a programmatic solution rather than a spreadsheet driven solution for modeling needs going forward
- Advantages include:
  - **Ease of extendibility**: New resources and activities can be easily added to
  - **Clarity of input parameters** : Various human readable formats can be easily used to describe and document each input parameter
  - **Not monolithic**: Easier to do some analysis, ie, Parameter tradeoff studies
  - **Version control and change tracking**: Model evolution and input parameter evolution can easily be tracked and shared
  - **Unit testing**: Model components can be more easily tested

Current Approach



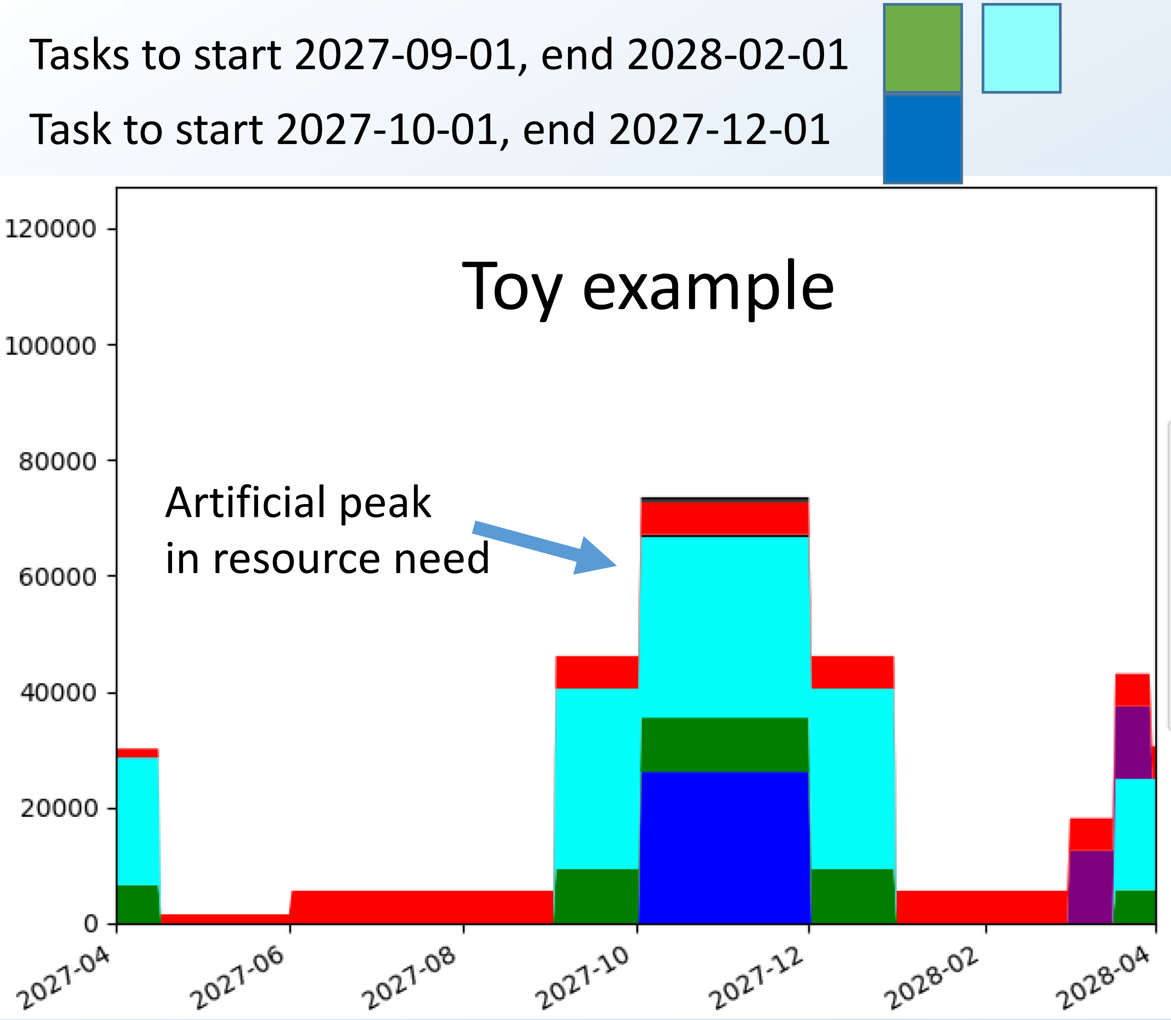
Approach and Implementation

Feature	Implications / Functionalities
Extendable set of resource types	Straightforward to add a dynamic resources, network model or cost estimates
Extendable set of data types	Easy to include evolutions in experiment computing model
Extendable set of activities	
Extendable set of parameterizations	Scaling with # of events, scaling with luminosity, etc
Easily scriptable	Straightforward implementation of tradeoff studies, plotting, etc using standard tools
	Easy to implement experiment ”policies”
Fidelity driven by configuration	Easy to control level of detail in tasks and plots

Workflow balancing in time and across resources

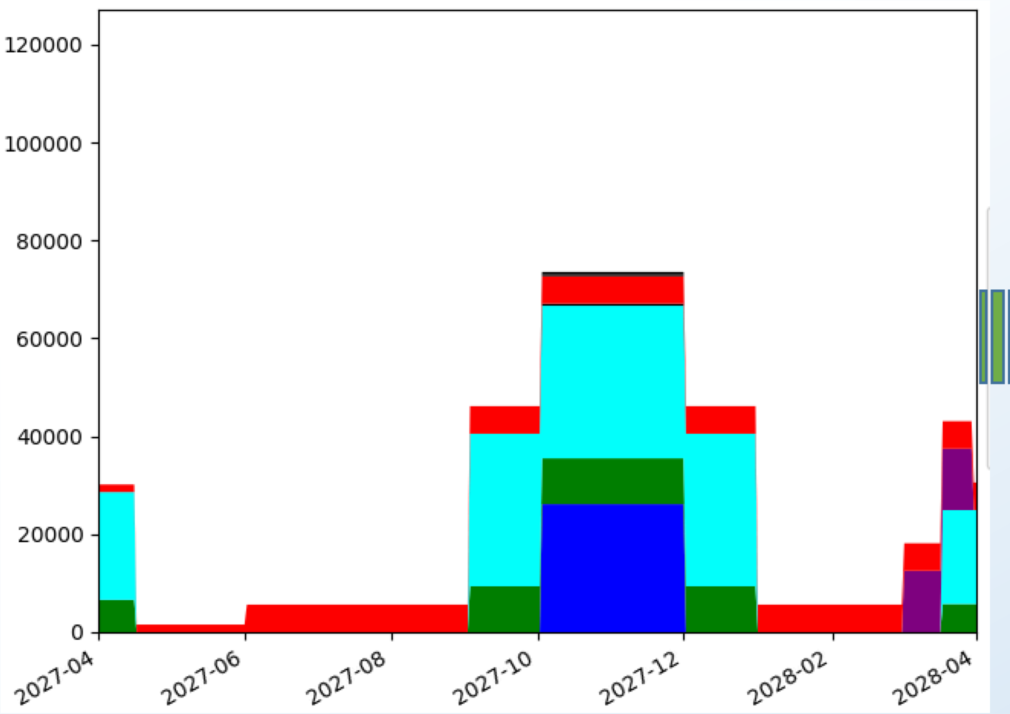
- Challenge:
  - Activities need to be performed in the time window between when the software is ready and when the events are needed for analysis
  - Models assume a flat resource set for processing tasks between these dates

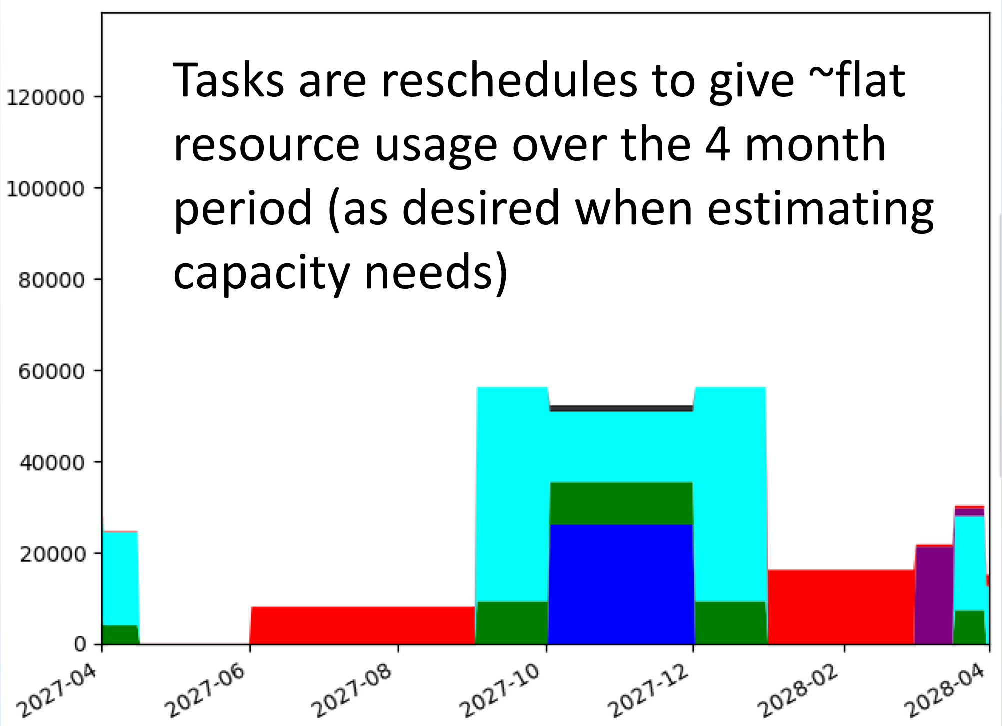
- Traditionally, the resource modeler is responsible for fine tuning to avoid artificial peaks/dips due to
  - Start / stop times per activity
  - Fraction of data processed / stored at each tier for each activity
  - Effects of resources available for part of the year



Activities can be automatically scheduled across time and resource types given constraints

- Constraints must include
  - Respect ”start by” and “finish by” dates
  - Not all activities are flexible (eg, analysis)

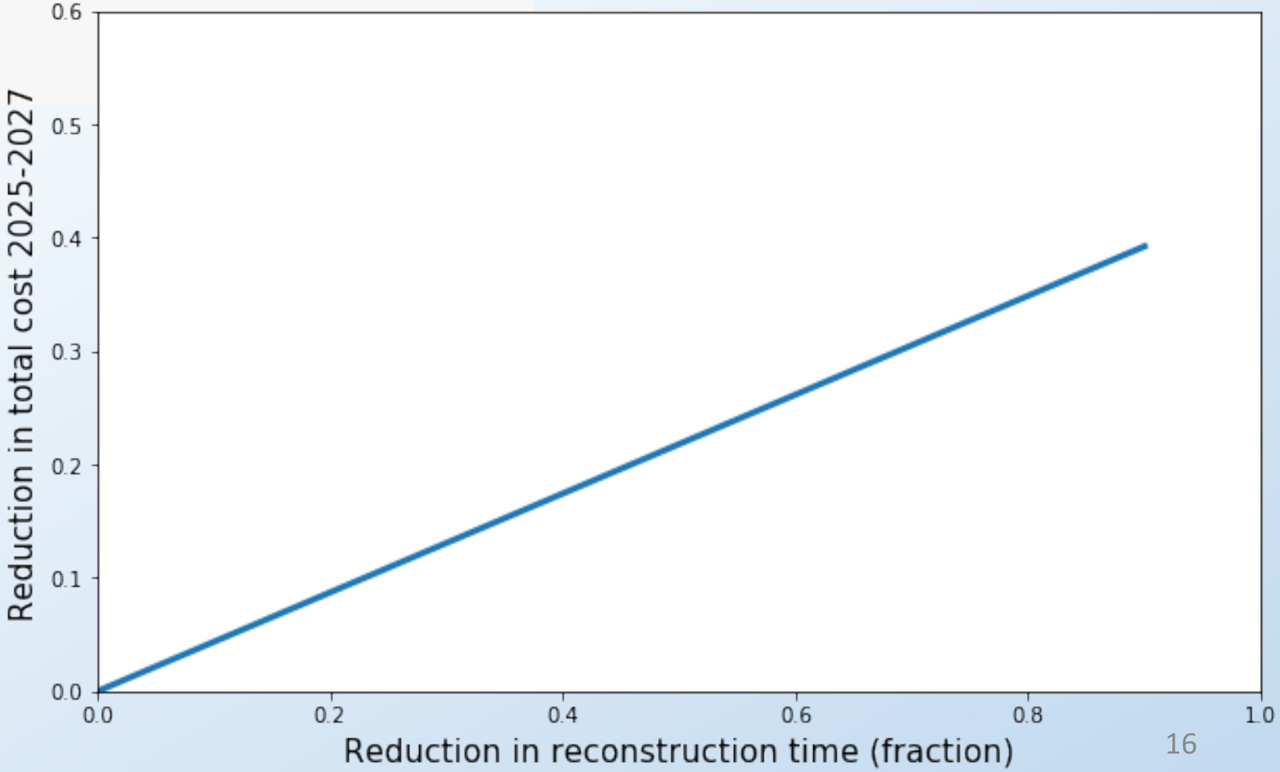




Example – cost savings for improving HL-LHC projections

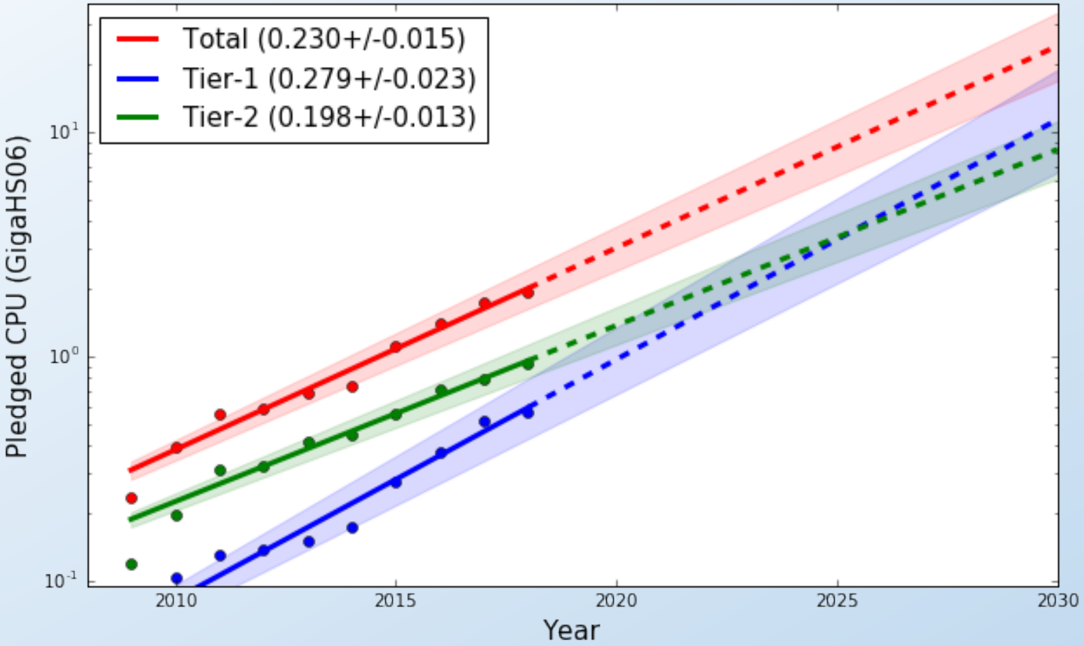
```
default_reco_hs06 = model['cpu_time']['data']['RECO']["2026"]
default_reco_hs06_mc = model['cpu_time']['mc']['RECO']["2026"]
vals = numpy.arange(1.0,0.0,-0.1)
costs=numpy.zeros(vals.shape[0])
for i,v in enumerate(vals):
    model['cpu_time']['data']['RECO']["2026"] = default_reco_hs06 * v
    model['cpu_time']['mc']['RECO']["2026"] = default_reco_hs06_mc * v
    metrics,details=run_computing_model(model)
    costs[i]=metrics['cost']
```

Parameter sensitivity studies can be easily performed against predefined metrics (e.g., total CPU, total "cost", etc)



Future challenges: Including uncertainties?

- Even before we consider **truly disruptive model changes**, model parameters have uncertainties (which are sometimes hard to estimate)
- These uncertainties should be propagated to ensure that model resources are framed in the correct context.
- For example
  - Resource extrapolations based on historical REBUS has 1-2% uncertainty **per year** of extrapolation
  - Achieving a 10% gain in code performance each year until HL-LHC instead of a 5% gain means **50% less** CPU needs



Conclusion

- New model for resource need projections in use in CMS for short-term and long-term studies and reviews
- Pieces of the structure are still evolving (as more test cases are considered). We hope to reach a releasable codebase this year