Building sustainable software communities

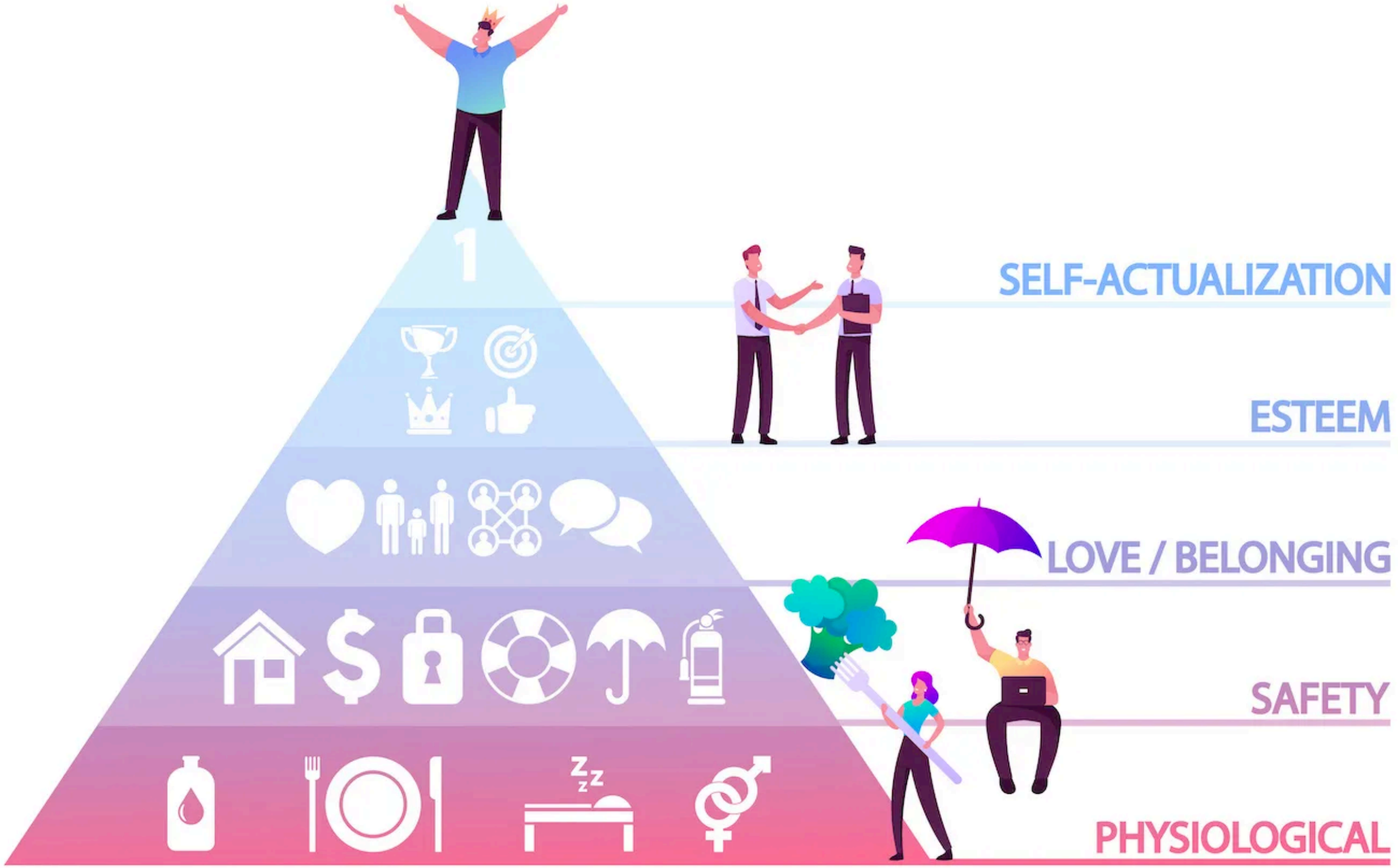
With thanks to many LHCb colleagues for input and being sounding boards for this talk

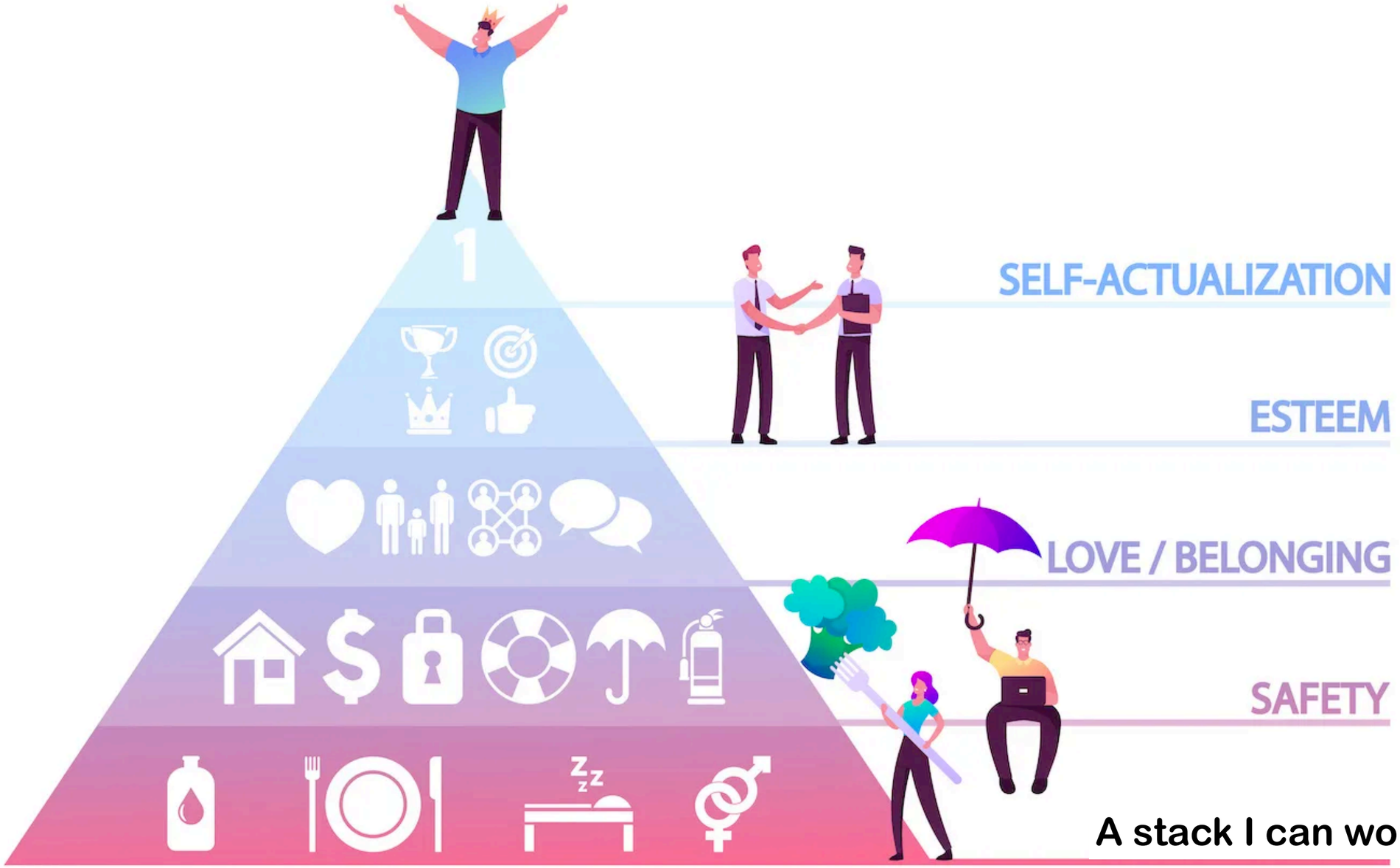
all bad/wrong opinions and errors are mine alone

Vladimir V. Gligorov, CNRS/LPNHE/CERN

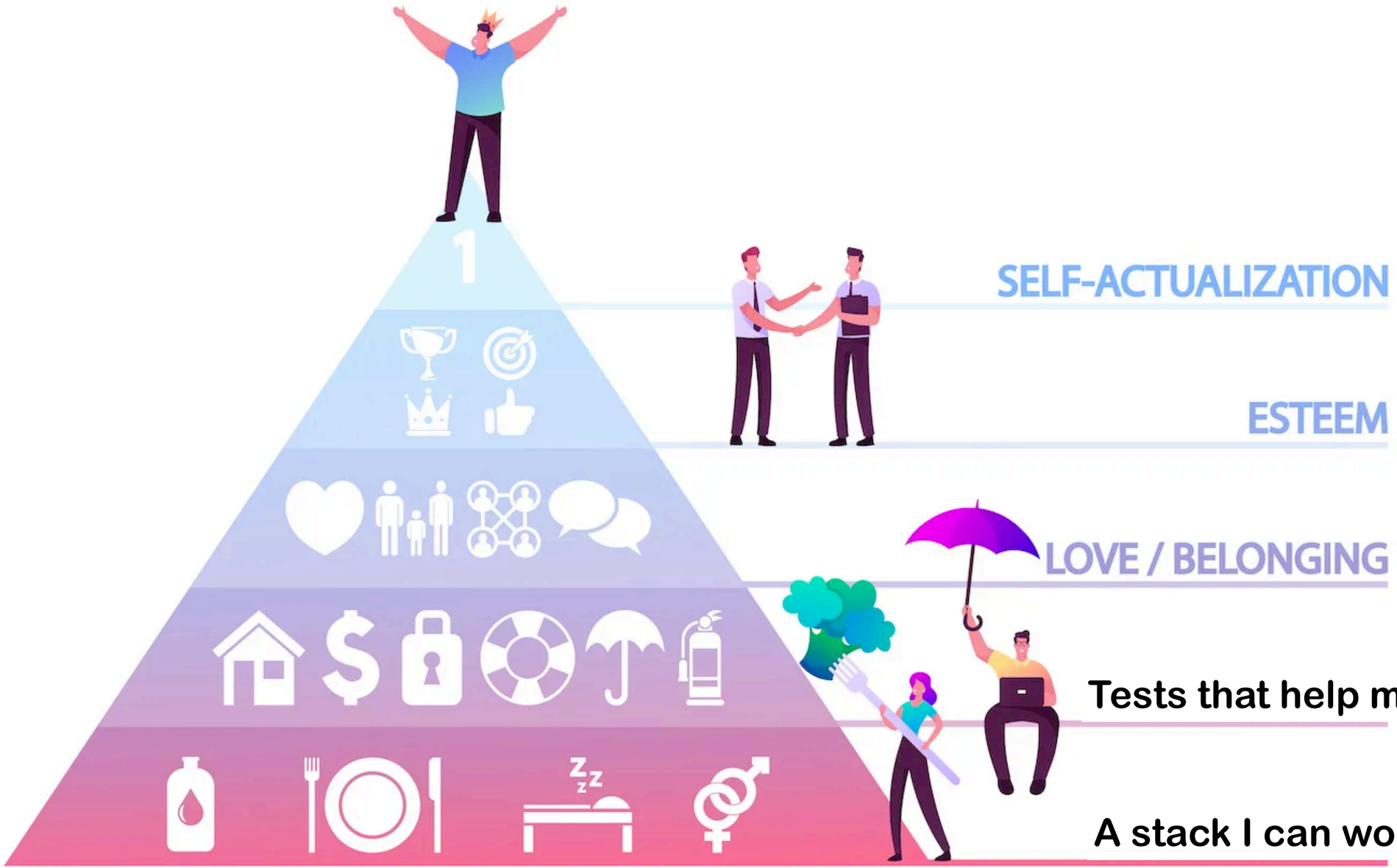
HSF Reconstruction & Software Triggers meeting

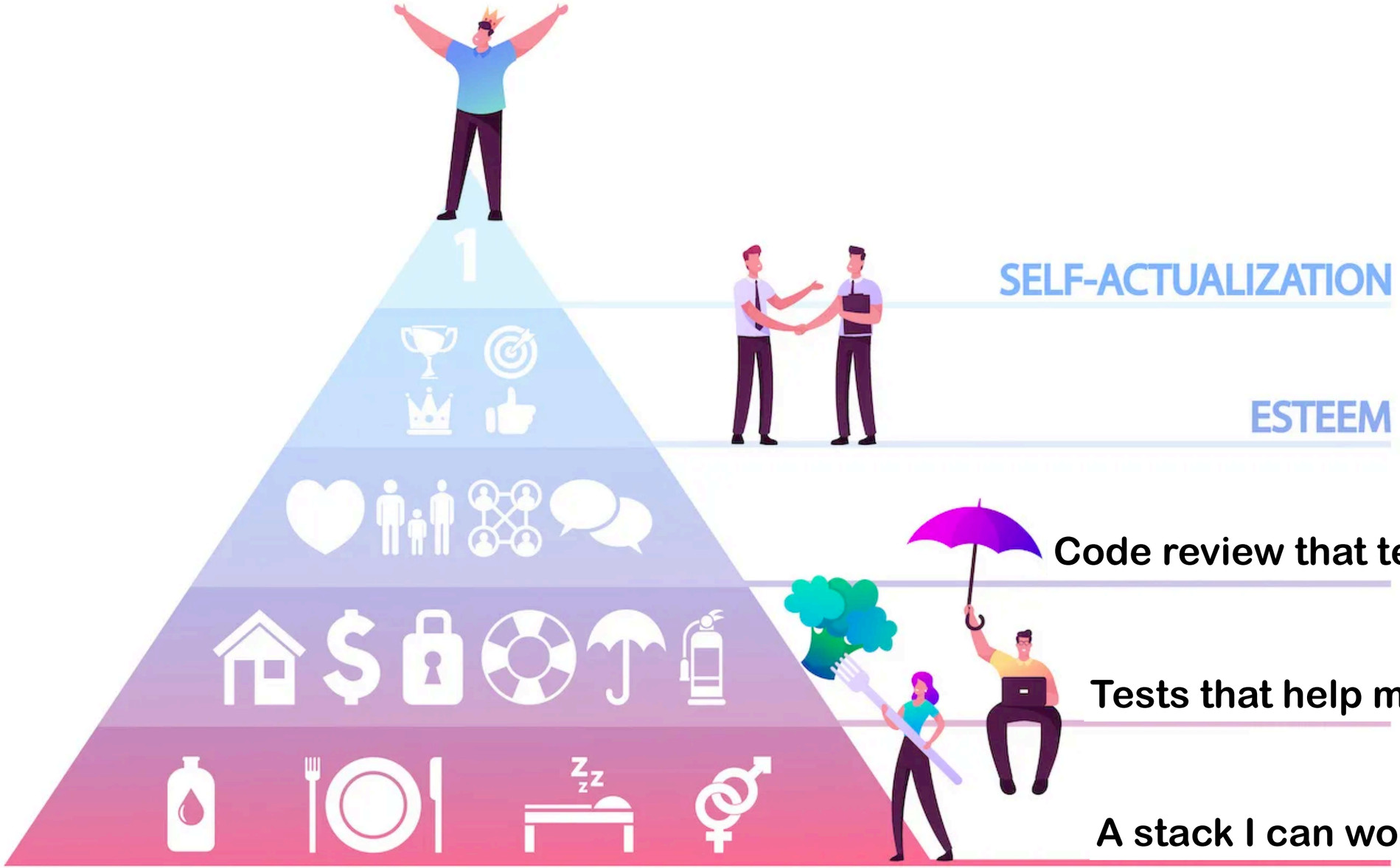
Cyberspace, 27.11.2024


1
SELF-ACTUALIZATION
ESTEEM
LOVE / BELONGING
SAFETY
PHYSIOLOGICAL

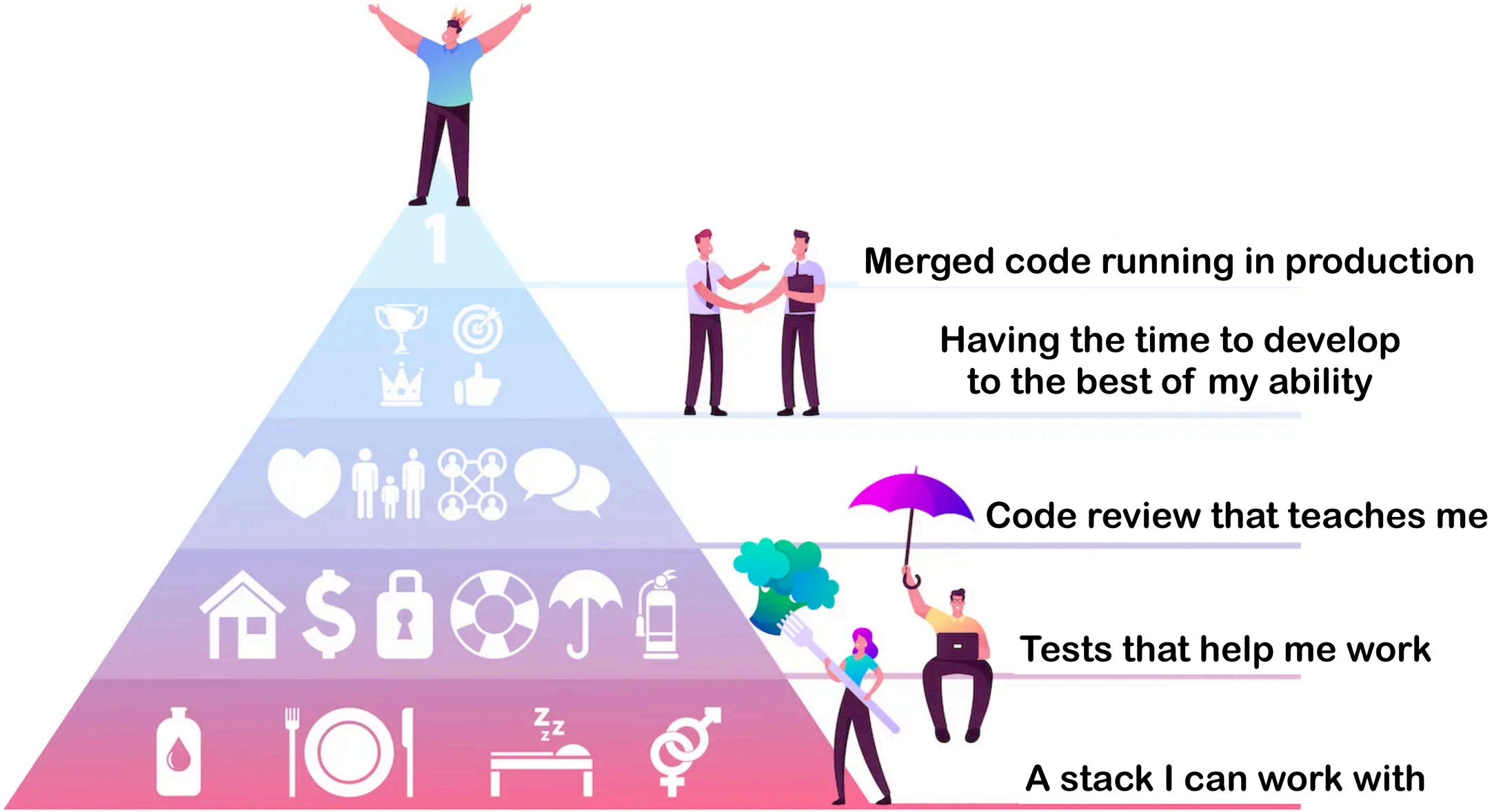


A stack I can work with


1
SELF-ACTUALIZATION
ESTEEM
LOVE / BELONGING
Tests that help me work
A stack I can work with


SELF-ACTUALIZATION
ESTEEM
Code review that teaches me
Tests that help me work
A stack I can work with


SELF-ACTUALIZATION
Having the time to develop to the best of my ability
Code review that teaches me
Tests that help me work
A stack I can work with


Merged code running in production
Having the time to develop
to the best of my ability
Code review that teaches me
Tests that help me work
A stack I can work with

General contextual axioms for this talk

1. A collaboration's software is a subdetector which is never turned off
2. This subdetector is constantly evolving
3. This subdetector is deployed in and must adjust to a constantly shifting environment
4. This subdetector must be maintained for years (decades) after it stops taking data

*One can nitpick that 2 and 3 apply to almost any system, but the point is the scale at which they apply

Shared training

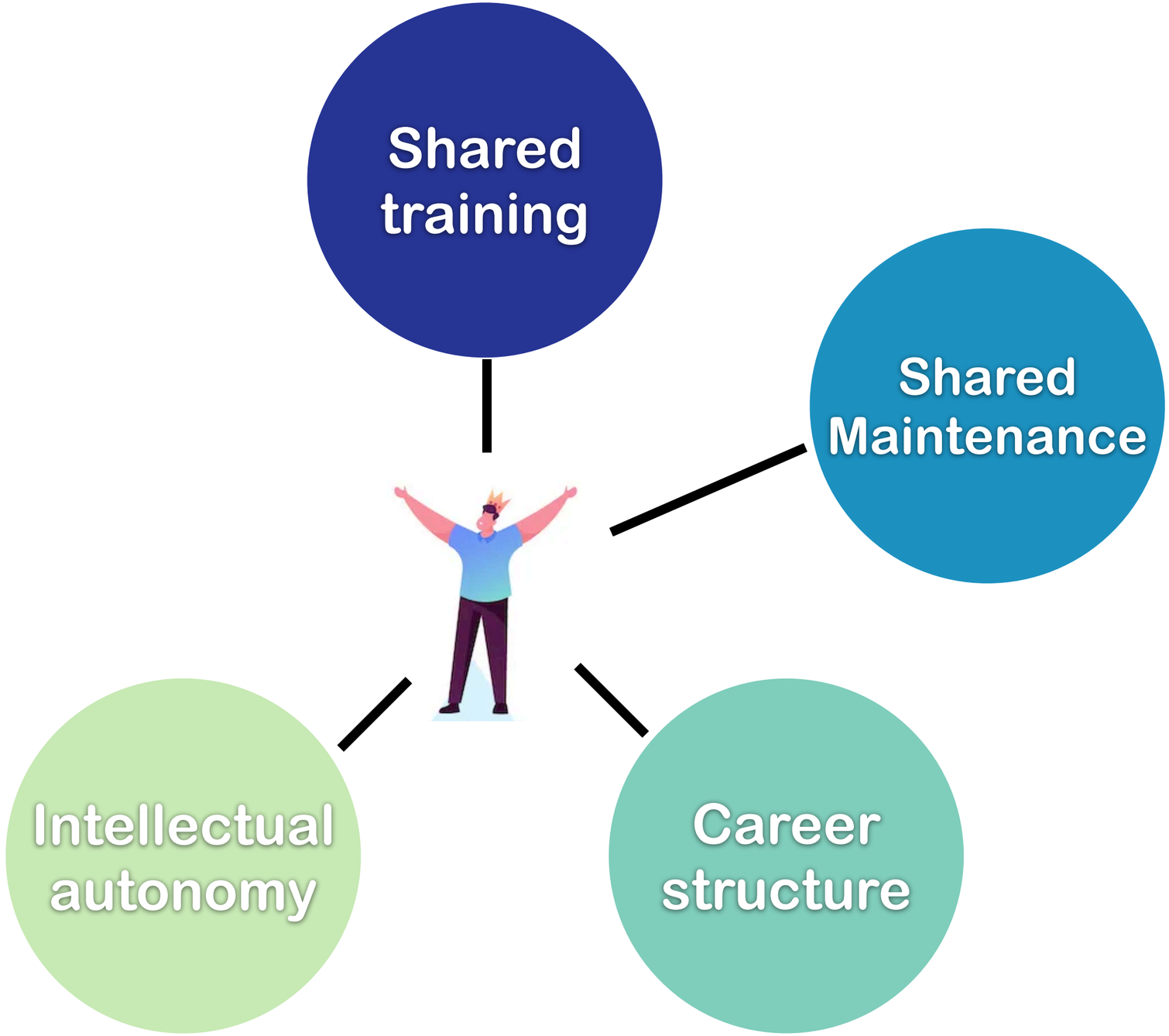
Shared training

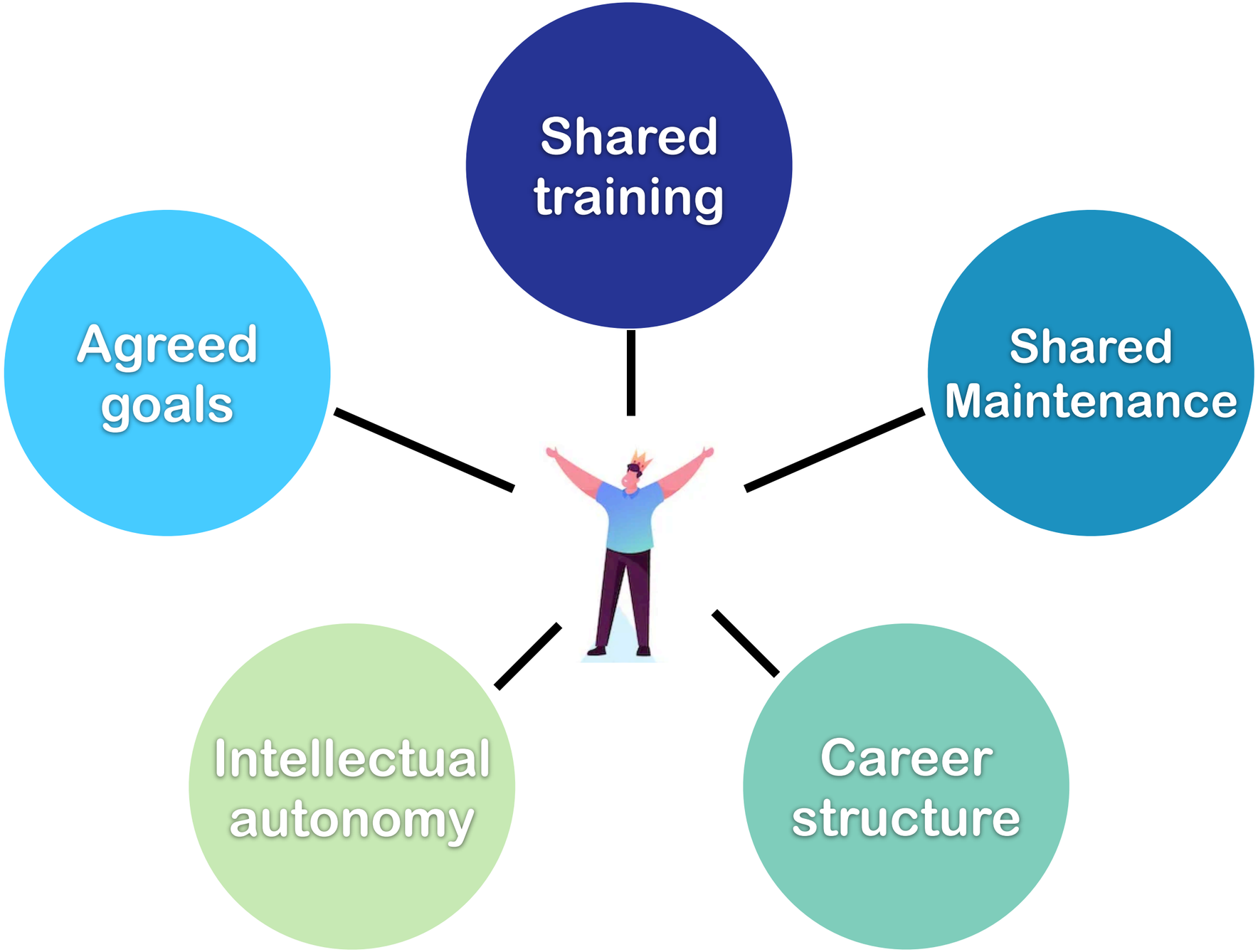
Shared Maintenance

Shared training

Shared Maintenance

Career structure


Shared training
Shared Maintenance
Intellectual autonomy
Career structure


Shared training
Agreed goals
Shared Maintenance
Intellectual autonomy
Career structure

Axioms concerning training

1. The requirements of real-time HEP software are at the edge of any application in any sector (including private)
2. Universities do not teach the skills needed to develop software at this level
3. Therefore we must teach them ourselves, while keeping up with developments outside HEP in architecture & languages

"SOA Particle" + combiner and filter to use it with

Merged **Niklas Stefan Nolte** requested to merge `NN_particlev2` into `master` 3 years ago

Overview 16 | Commits 54 | Pipelines 8 | Changes 22

depends on LHCb!2309 (merged) and Rec!1728 (merged)
DISCLAIMER: this does not aim to provide anything feature complete, rather the opposite. its super incomplete.

this supersedes !603 (closed) by modernizing the combiner introduced there.

other than that, i introduce some stuff that could at some point end up as variant alternative in `v2::Particle` for now, all of the things related to the model are stored in `Particle_v2.h`, although we will want to change that later (as it grows).

the `ChargedBasics` are currently a zip of many smaller things like `MuonPIDs`, `RichPIDs`, `ParticleIDs` and (for now) `Fitted::Forward::Tracks` (we probably want to change that at some point)

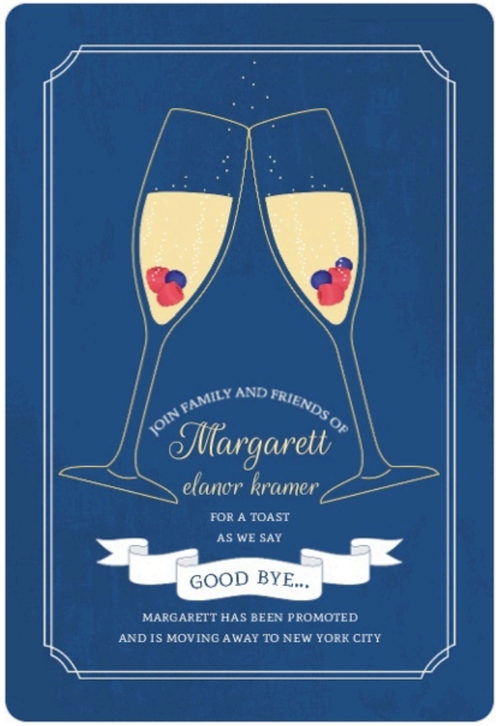
to fill the `ChargedBasics`, i currently run something that converts `LHCb::ProtoParticle`s to `ChargedBasics` (Particle_to_Particle_v2.h) (the protos come from file in the hlt2_example in moore)

the `ThOrParticleCombiner` currently supports 2,3,4 body combination, and the only possible signature right now is `[ChargedBasics] -> Composites` (also residing in Particle_v2.h). there are many open todos, like supporting combination of composites, vertex fitting with something other than the Closest To Beam state, vectorizing the combination cut and so on. a (incomplete) list of todos is listed in the combiner file, together with `//TODO`s here and there

there are significant speedups to be gained, the ones i saw during development ranged from 2x to 10x (do not hold me accountable or quote me for these numbers in the end), even with a fast vertexfitter (`ParticleVertexFitter`) and distancecalculator `TrgDistanceCalculator` activated.

you may ask: why merge this if its so highly incomplete? because this MR is already huge and incremental steps seem better for synchronization with others and to try to keep an overview. and rebasing is a PITA

thx to @olupton for the help with debugging :)



"SOA Particle" + combiner and filter to use it with

Merged **Niklas Stefan Nolte** requested to merge `NN_particlev2` into `master` 3 years ago

Overview 16 | Commits 54 | Pipelines 8 | Changes 22

depends on LHCb!2309 (merged) and Rec!1728 (merged)
DISCLAIMER: this does not aim to provide anything feature complete, rather the opposite. its super incomplete.

this supersedes !603 (closed) by modernizing the combiner introduced there.

other than that, i introduce some stuff that could at some point end up as variant alternative in `v2::Particle` for now, all of the things related to the model are stored in `Particle_v2.h`, although we will want to change that later (as it grows).

the `ChargedBasics` are currently a zip of many smaller things like `MuonPIDs`, `RichPIDs`, `ParticleIDs` and (for now) `Fitted::Forward::Tracks` (we probably want to change that at some point)

to fill the `ChargedBasics`, i currently run something that converts `LHCb::ProtoParticle`s to `ChargedBasics` (Particle_to_Particle_v2.h) (the protos come from file in the hlt2_example in moore)

the `ThOrParticleCombiner` currently supports 2,3,4 body combination, and the only possible signature right now is `[ChargedBasics] -> Composites` (also residing in Particle_v2.h). there are many open todos, like supporting combination of composites, vertex fitting with something other than the Closest To Beam state, vectorizing the combination cut and so on. a (incomplete) list of todos is listed in the combiner file, together with `//TODO`s here and there

there are significant speedups to be gained, the ones i saw during development ranged from 2x to 10x (do not hold me accountable or quote me for these numbers in the end), even with a fast vertexfitter (`ParticleVertexFitter`) and distancecalculator `TrgDistanceCalculator` activated.

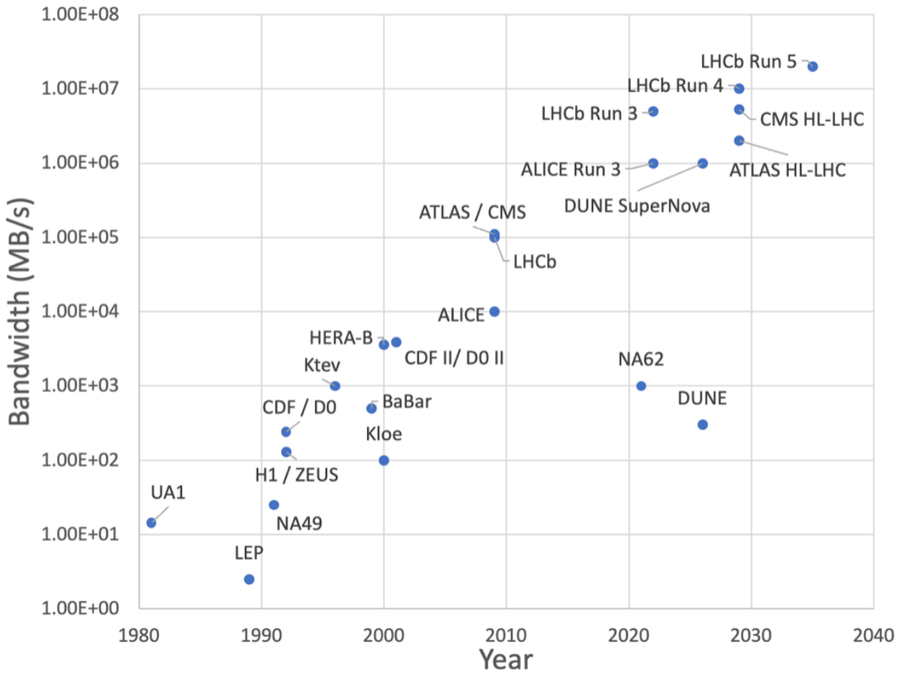
you may ask: why merge this if its so highly incomplete? because this MR is already huge and incremental steps seem better for synchronization with others and to try to keep an overview. and rebasing is a PITA

thx to @olupton for the help with debugging :)

Navigating a heterogeneous world

The volume of data processed in real-time has increased by 1-2 orders of magnitude per decade with little sign of slowing down

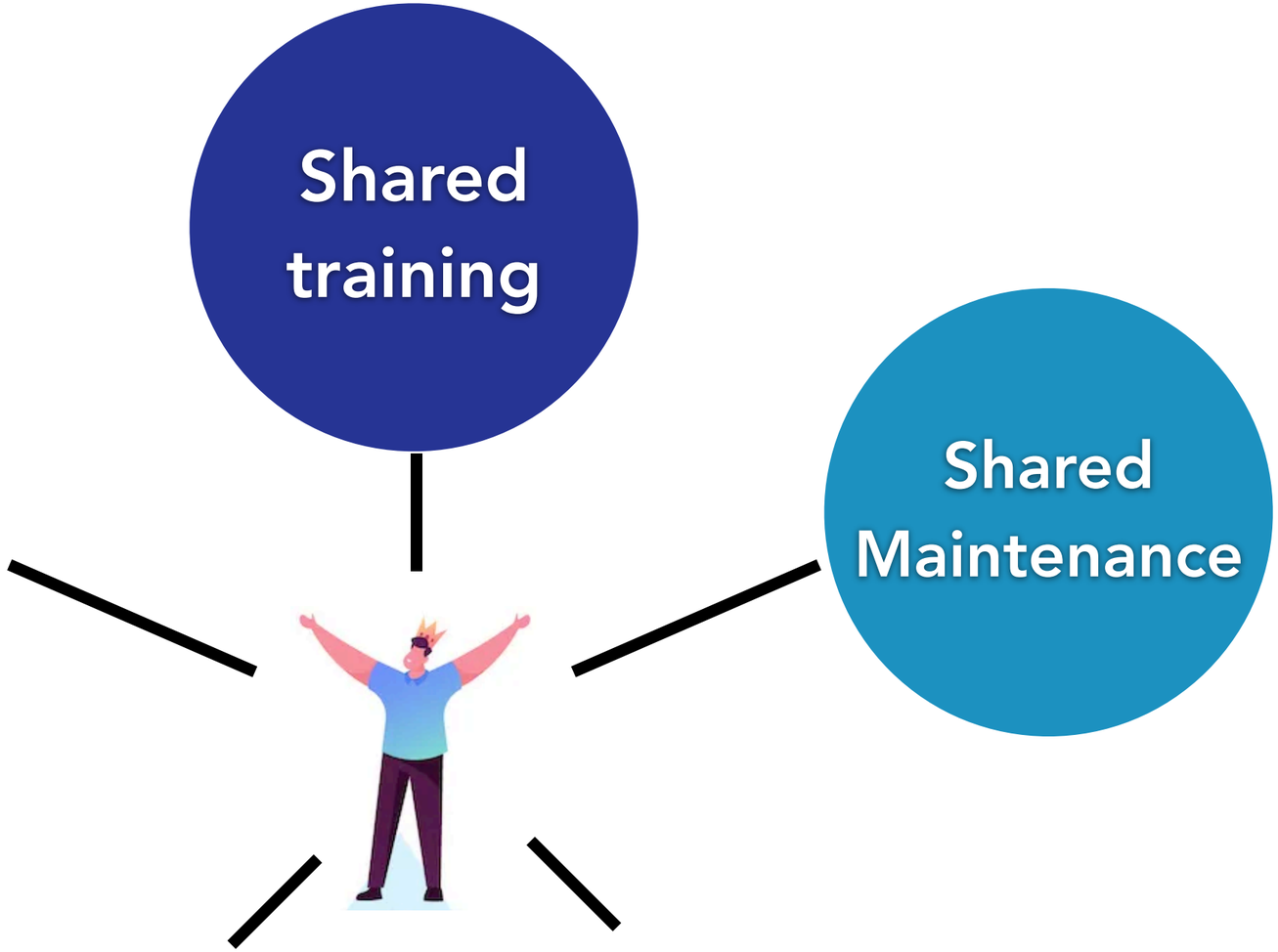
When figuring out how to best utilise existing and future scientific facilities we are challenged by the ongoing fragmentation of high-performance computing architectures, as well as computing languages



Atomic projects addressing one part of the processing pipeline offer an attractive way to train and retain talented people because they can offer both a manageable learning curve and concrete career payoffs.

On the other hand integrating these into production systems and mantaining them can bring significant hidden costs.

Balancing R&D and production is intimately connected to how we prioritise training and heavily influences people's careers. It is a key challenge facing in particular the trigger community over the next decade(s).


Shared training
Shared Maintenance


SELF-ACTUALIZATION
ESTEEM
Code review that teaches me
Tests that help me work
A stack I can work with
MAINTENANCE

"SOA Particle" + combiner and filter to use it with

Merged **Niklas Stefan Nolte** requested to merge `NN_particlev2` into `master` 3 years ago

Overview 16 | Commits 54 | Pipelines 8 | Changes 22

depends on LHCb!2309 (merged) and Rec!1728 (merged)
DISCLAIMER: this does not aim to provide anything feature complete, rather the opposite. its super incomplete.

this supersedes !603 (closed) by modernizing the combiner introduced there.

other than that, i introduce some stuff that could at some point end up as variant alternative in `v2::Particle` for now, all of the things related to the model are stored in `Particle_v2.h`, although we will want to change that later (as it grows).

the `ChargedBasics` are currently a zip of many smaller things like `MuonPIDs`, `RichPIDs`, `ParticleIDs` and (for now) `Fitted::Forward::Tracks` (we probably want to change that at some point)

to fill the `ChargedBasics`, i currently run something that converts `LHCb::ProtoParticle`s to `ChargedBasics` (Particle_to_Particle_v2.h) (the protos come from file in the hlt2_example in moore)

the `ThOrParticleCombiner` currently supports 2,3,4 body combination, and the only possible signature right now is `[ChargedBasics] -> Composites` (also residing in Particle_v2.h). there are many open todos, like supporting combination of composites, vertex fitting with something other than the Closest To Beam state, vectorizing the combination cut and so on. a (incomplete) list of todos is listed in the combiner file, together with `//TODO`s here and there

there are significant speedups to be gained, the ones i saw during development ranged from 2x to 10x (do not hold me accountable or quote me for these numbers in the end), even with a fast vertexfitter (`ParticleVertexFitter`) and distancecalculator `TrgDistanceCalculator` activated.

you may ask: why merge this if its so highly incomplete? because this MR is already huge and incremental steps seem better for synchronization with others and to try to keep an overview. and rebasing is a PITA

thx to @olupton for the help with debugging :)

"SOA Particle" + combiner and filter to use it with

Merged **Niklas Stefan Nolte** requested to merge `NN_particlev2` into `master` 3 years ago

Overview 16 | Commits 54 | Pipelines 8 | Changes 22

depends on LHCb!2309 (merged) and Rec!1728 (merged)
DISCLAIMER: this does not aim to provide anything feature complete, rather the opposite. its super incomplete.

this supersedes !603 (closed) by modernizing the combiner introduced there.

other than that, i introduce some stuff that could at some point end up as variant alternative in `v2::Particle` for now, all of the things related to the model are stored in `Particle_v2.h`, although we will want to change that later (as it grows).

the `ChargedBasics` are currently a zip of many smaller things like `MuonPIDs`, `RichPIDs`, `ParticleIDs` and (for now) `Fitted::Forward::Tracks` (we probably want to change that at some point)

to fill the `ChargedBasics`, i currently run something that converts `LHCb::ProtoParticle`s to `ChargedBasics` (Particle_to_Particle_v2.h) (the protos come from file in the hlt2_example in moore)

the `ThOrParticleCombiner` currently supports 2,3,4 body combination, and the only possible signature right now is `[ChargedBasics] -> Composites` (also residing in Particle_v2.h). there are many open todos, like supporting combination of composites, vertex fitting with something other than the Closest To Beam state, vectorizing the combination cut and so on. a (incomplete) list of todos is listed in the combiner file, together with `//TODO`s here and there

there are significant speedups to be gained, the ones i saw during development ranged from 2x to 10x (do not hold me accountable or quote me for these numbers in the end), even with a fast vertexfitter (`ParticleVertexFitter`) and distancecalculator `TrgDistanceCalculator` activated.

you may ask: why merge this if its so highly incomplete? because this MR is already huge and incremental steps seem better for synchronization with others and to try to keep an overview. and rebasing is a PITA

thx to @olupton for the help with debugging :)

Maintenance and training go together

1. It is in the interest of all developers to keep the codebase “clean” and tests green
2. Even more so for a real-time system!
3. If you spend time training people, you want them to help with the maintenance.
4. But physicists are often actively penalised for working on maintenance

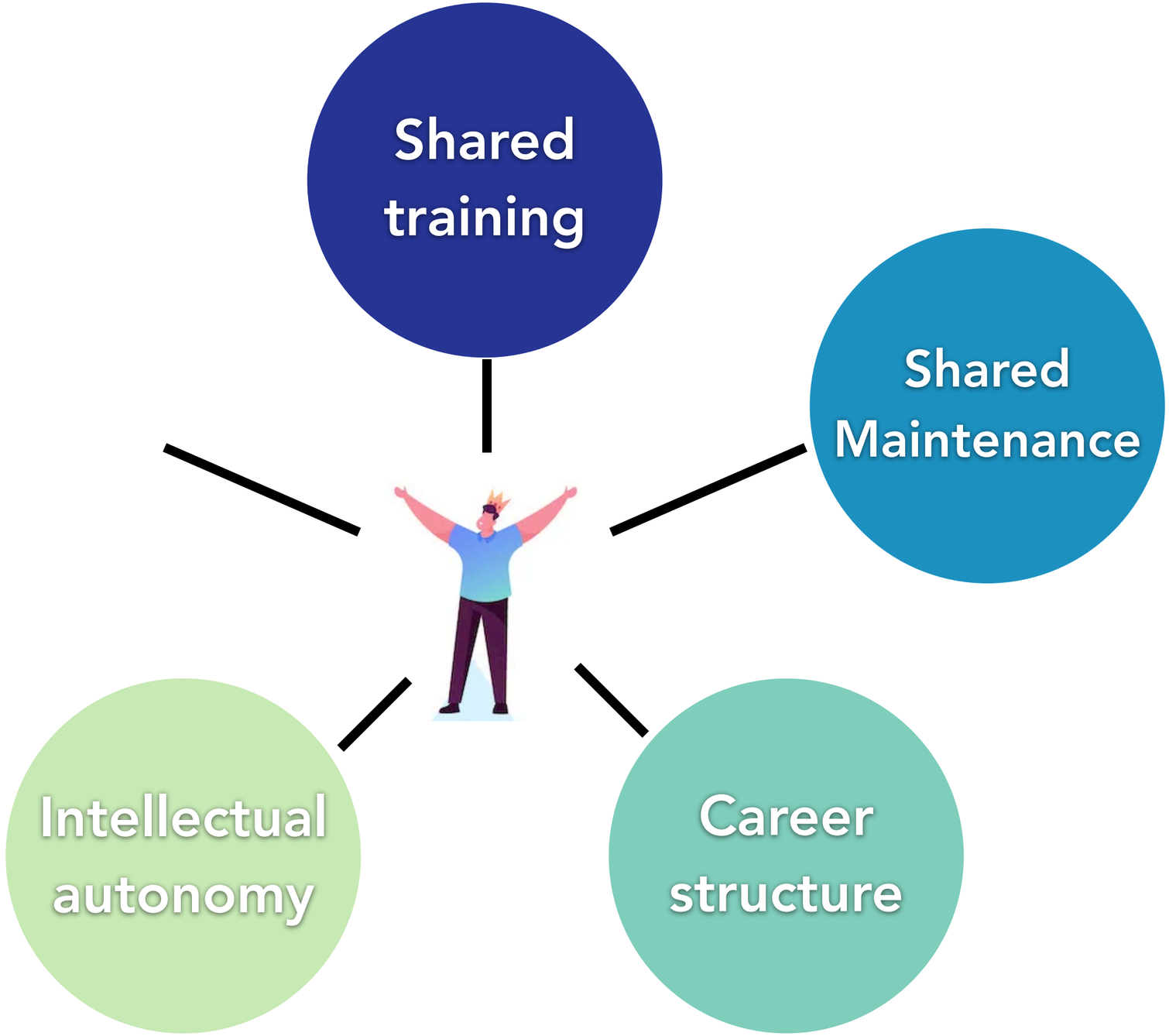
RTA Software Shifter

« ‹ › »

July 2023

today

Mon	Tue	Wed	Thu	Fri	Sat	Sun
26 Ao Xu	27 Ao Xu	28 Ao Xu	29 Ao Xu	30 Ao Xu	1 Ao Xu	2 Ao Xu
3 Ao Xu	4 Ao Xu	5 Ao Xu	6 Ao Xu	7 Ao Xu	8 Ao Xu	9 Ao Xu
10 Milosz Jerzy Zdybal	11 Milosz Jerzy Zdybal	12 Milosz Jerzy Zdybal	13 Milosz Jerzy Zdybal	14 Milosz Jerzy Zdybal	15 Milosz Jerzy Zdybal	16 Milosz Jerzy Zdybal
17 Milosz Jerzy Zdybal	18 Milosz Jerzy Zdybal	19 Milosz Jerzy Zdybal	20 Milosz Jerzy Zdybal	21 Milosz Jerzy Zdybal	22 Milosz Jerzy Zdybal	23 Milosz Jerzy Zdybal
24 Jamie Gooding	25 Jamie Gooding	26 Jamie Gooding	27 Jamie Gooding	28 Jamie Gooding	29 Jamie Gooding	30 Jamie Gooding
31 Jamie Gooding	1 Jamie Gooding	2 Jamie Gooding	3 Jamie Gooding	4 Jamie Gooding	5 Jamie Gooding	6 Jamie Gooding


Shared training
Shared Maintenance
Intellectual autonomy
Career structure

"SOA Particle" + combiner and filter to use it with

Merged **Niklas Stefan Nolte** requested to merge `NN_particlev2` into `master` 3 years ago

Overview 16 | Commits 54 | Pipelines 8 | Changes 22

depends on LHCb!2309 (merged) and Rec!1728 (merged)
DISCLAIMER: this does not aim to provide anything feature complete, rather the opposite. its super incomplete.

this supersedes !603 (closed) by modernizing the combiner introduced there.

other than that, i introduce some stuff that could at some point end up as variant alternative in `v2::Particle` for now, all of the things related to the model are stored in `Particle_v2.h`, although we will want to change that later (as it grows).

the `ChargedBasics` are currently a zip of many smaller things like `MuonPIDs`, `RichPIDs`, `ParticleIDs` and (for now) `Fitted::Forward::Tracks` (we probably want to change that at some point)

to fill the `ChargedBasics`, i currently run something that converts `LHCb::ProtoParticle`s to `ChargedBasics` (Particle_to_Particle_v2.h) (the protos come from file in the hlt2_example in moore)

the `ThOrParticleCombiner` currently supports 2,3,4 body combination, and the only possible signature right now is `[ChargedBasics] -> Composites` (also residing in Particle_v2.h). there are many open todos, like supporting combination of composites, vertex fitting with something other than the Closest To Beam state, vectorizing the combination cut and so on. a (incomplete) list of todos is listed in the combiner file, together with `//TODO`s here and there

there are significant speedups to be gained, the ones i saw during development ranged from 2x to 10x (do not hold me accountable or quote me for these numbers in the end), even with a fast vertexfitter (`ParticleVertexFitter`) and distancecalculator `TrgDistanceCalculator` activated.

you may ask: why merge this if its so highly incomplete? because this MR is already huge and incremental steps seem better for synchronization with others and to try to keep an overview. and rebasing is a PITA

thx to @olupton for the help with debugging :)

Rolling contracts

Teaching & admin

Chasing grant money

"SOA Particle" + combiner and filter to use it with

Merged **Niklas Stefan Nolte** requested to merge `NN_particlev2` into `master` 3 years ago

Overview 16 | Commits 54 | Pipelines 8 | Changes 22

depends on LHCb!2309 (merged) and Rec!1728 (merged)
DISCLAIMER: this does not aim to provide anything feature complete, rather the opposite. its super incomplete.

this supersedes !603 (closed) by modernizing the combiner introduced there.

other than that, i introduce some stuff that could at some point end up as variant alternative in `v2::Particle` for now, all of the things related to the model are stored in `Particle_v2.h`, although we will want to change that later (as it grows).

the `ChargedBasics` are currently a zip of many smaller things like `MuonPIDs`, `RichPIDs`, `ParticleIDs` and (for now) `Fitted::Forward::Tracks` (we probably want to change that at some point)

to fill the `ChargedBasics`, i currently run something that converts `LHCb::ProtoParticle`s to `ChargedBasics` (Particle_to_Particle_v2.h) (the protos come from file in the hlt2_example in moore)

the `ThOrParticleCombiner` currently supports 2,3,4 body combination, and the only possible signature right now is `[ChargedBasics] -> Composites` (also residing in Particle_v2.h). there are many open todos, like supporting combination of composites, vertex fitting with something other than the Closest To Beam state, vectorizing the combination cut and so on. a (incomplete) list of todos is listed in the combiner file, together with `//TODO`s here and there

there are significant speedups to be gained, the ones i saw during development ranged from 2x to 10x (do not hold me accountable or quote me for these numbers in the end), even with a fast vertexfitter (`ParticleVertexFitter`) and distancecalculator `TrgDistanceCalculator` activated.

you may ask: why merge this if its so highly incomplete? because this MR is already huge and incremental steps seem better for synchronization with others and to try to keep an overview. and rebasing is a PITA

thx to @olupton for the help with debugging :)

Misaligned incentives in HEP SW jobs

1. Some countries are creating “technology-oriented” positions, but typically cross-experiment and R&D oriented
2. Long-term support for physicists to start groups around software remains poor
3. The money being spent generally targets ML/AI applications which sometimes distorts incentives for production code

Implementation of the ECFA Detector R&D Roadmap

After the publication of the ECFA Detector R&D Roadmap, CERN Council requested ECFA to develop the plan for its implementation.

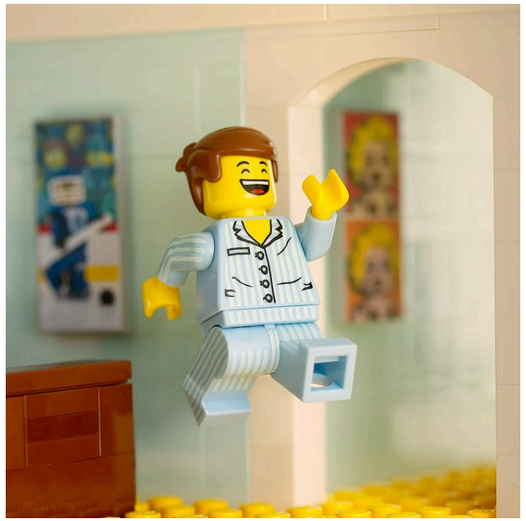
The document approved by the SPC and CERN Council in September 2022 can be found at https://indico.cern.ch/event/1197445/contributions/5034860/attachments/2517863/4329123/spc-e-1190-c-e-3679-Implementation_Detector_Roadmap.pdf.

As proposed in the document, topic specific community meetings have been held throughout 2023 with sign up for more information in the links listed below.

- TF1 Gaseous Detectors https://indico.cern.ch/event/1214405/
- TF2 Liquid Detectors https://indico.cern.ch/event/1214404/
- TF3 Solid State Detectors https://indico.cern.ch/event/1214410/
- TF4 Photon Detectors and PID https://indico.cern.ch/event/1214407/
- TF5 Quantum and Emerging Technologies https://indico.cern.ch/event/1214411/
- TF6 Calorimetry https://indico.cern.ch/event/1213733/
- TF7 Electronics and On-detector Processing https://indico.cern.ch/event/1214423/
- TF8 Integration https://indico.cern.ch/event/1214428/
- TF9 Training https://indico.cern.ch/event/1214429/

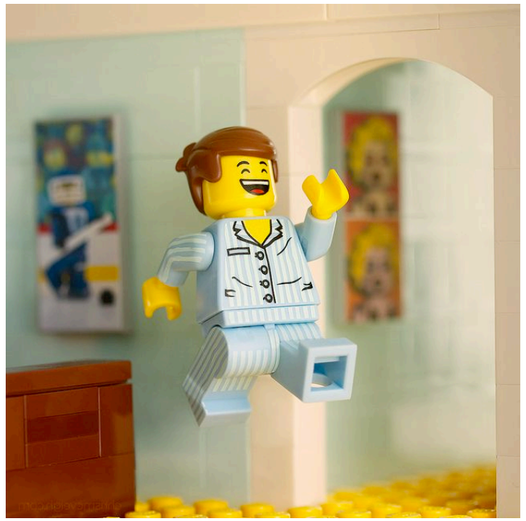
The reviewing process for the proposals is now well underway, with the DRDC now set up at https://committees.web.cern.ch/drdc supported by the ECFA Detector Panel (https://ecfa-dp.desy.de).

Low pay



Low pay

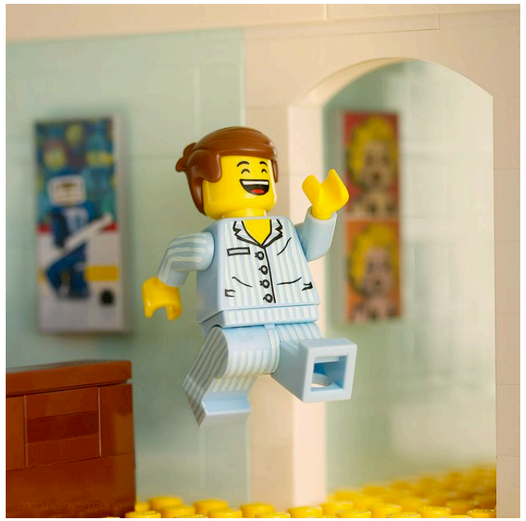
Insecure and opaque career path



Low pay

Insecure and opaque career path

Decade-long projects

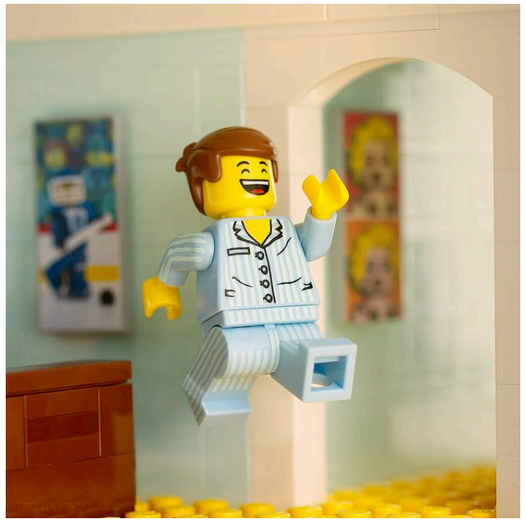


Low pay

Insecure and opaque career path

Decade-long projects

No clear path beyond SM



Low pay

Insecure and opaque career path

Decade-long projects

No clear path beyond SM

Excellent pay

Few-year projects

Fairly clear career

Answer big questions

Whether or not this is a fair description of reality, what matters is how it appears to students!

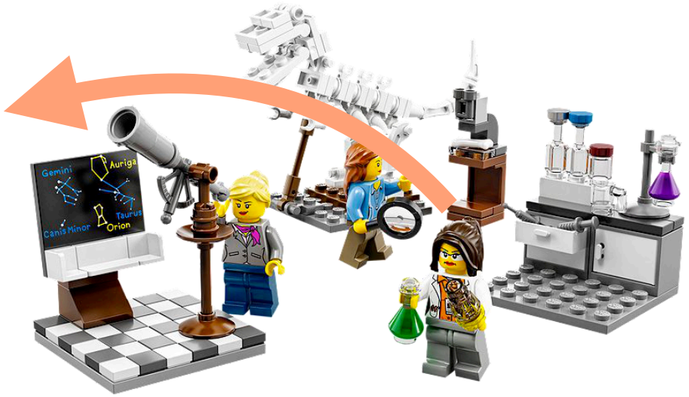
Scoping is important

1. HEP operates a high staff turnover model, and we have little chance to change this
2. Salaries are generally uncompetitive and again there is little chance to change this
3. Where we have some hope is tuning our enviroment to maximize the chances of career breakthroughs for early career staff


Gemini
Auriga
Canis Minor
Taurus
Orion


Gemini
Auriga
Canis Minor
Taurus
Orion


Gemini
Auriga
Canis Minor
Taurus
Orion


Gemini
Auriga
Canis Minor
Taurus
Orion

Is recognition a trap?

Discussions around software careers inevitably loop back to the topic of increased recognition

However, asking for recognition makes you dependent on the opinions of others

In addition we often ask for recognition when what we really mean is "I should be able to work on the things I believe are important without worrying about my career safety"

This is why I advocate that it is better to directly ask for autonomy and the resources to pursue it

Shared training

Agreed goals

Shared Maintenance

Intellectual autonomy

Career structure

Last update: April 2023

My own unofficial interpretation of the LHC schedule relevant for software!

Final thoughts on training & careers

1. **Creating sustainable software communities is one of the biggest challenges facing the field.**

Final thoughts on training & careers

1. Creating sustainable software communities is one of the biggest challenges facing the field.
2. Technology can't give us a value system, but it can make certain prioritisations easier.

Final thoughts on training & careers

1. Creating sustainable software communities is one of the biggest challenges facing the field.
2. Technology can't give us a value system, but it can make certain prioritisations easier.
3. The world around us increasingly recognises that performant and reliable software is key to desirable hardware. So should we!

Final thoughts on training & careers

1. Creating sustainable software communities is one of the biggest challenges facing the field.
2. Technology can't give us a value system, but it can make certain prioritisations easier.
3. The world around us increasingly recognises that performant and reliable software is key to desirable hardware. So should we!
4. A high-turnover staffing model is painful and requires continuous vigilance and a focus on individual progression to not be exploitative.

BACKUP

The eternal cycle of software training

1. Training new developers is expensive
2. For C++/performance code many institutes have little training capacity
3. Remaining effective post-training requires you to use the acquired skills
4. Unfortunately once trained, people often have to move to analysis or hardware work in order to get tenure

*All these issues are structural and none of it is the fault of the people being trained