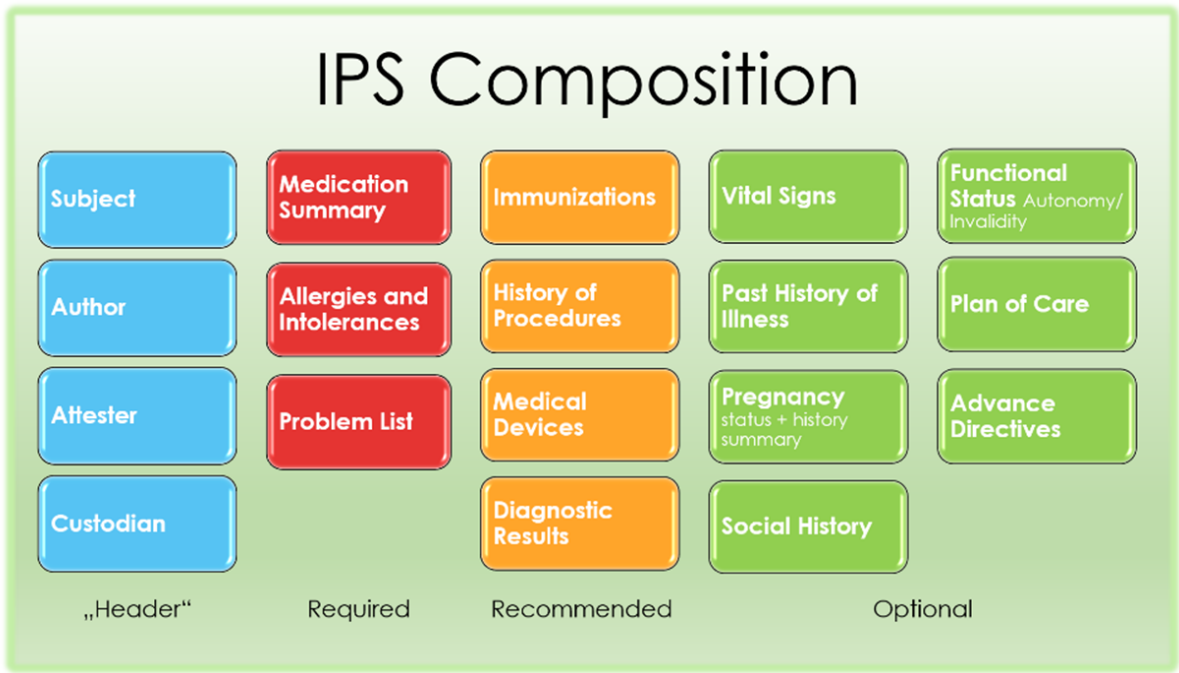


GELLO and the FHIR4 based International Patient Summary (IPS)

As has been stated previously GELLO is model agnostic. The International Patient Summary (IPS) is a health summary structure for patients who may need health care in another country.

It has the usual resources according to the '80/20 rule' .

The following diagram is taken from the home page for it located at <https://hl7.org/fhir/uv/ips/ipsStructure.html>



It is an ISO standard avail in Australia as **AS ISO 27269:2022**

A gello model can be made compatible with it and the FHIR4 structure it inherits. This in turn allows CDS script to run against IPS files.

Instance data has been built for the purposes of this demonstration but the model and the data are not shown here in the interests of brevity.

The following two examples show the GELLO editor script and results for two gello file executions.

First lets just see a text based view of the data, returned as a tuple:

Medical-Objects GELLO IDE (Mowgli)

File Edit Search View Run Build Tools Help

runFailure_IPS.gello test.gello

```
1 imports FHIR4 /* International Patient Summary */
2 Context Bundle
3
4 Let patient: Patient = entry.resource.Patient->flatten().first()
5 Let allMedications: Sequence(Medication) = entry.resource.Medication->flatten()
6 Let medCodes: Sequence(Coding) = entry.resource.Medication.code.coding->flatten()
7 Let medMeth: Sequence(Coding) = medCodes->select(c[c.system.value="urn:oid:2.16.840.1.113883.2.4.4.1"])
8 Let medSCT: Sequence(Coding) = medCodes->select(c[c.system.value="http://snomed.info/sct"])
9 Let allergies: Sequence(AllergyIntolerance) = entry.resource.AllergyIntolerance->flatten()
10 Let problemList: Sequence(Condition) = entry.resource.Condition->flatten()
11 Let observations: Sequence(Observation) = entry.resource.Observation->flatten()
12 Let IPSSummary = Tuple{
13   patientDetails = patient.name.given->flatten().first().value + ' ' + patient.name.family.first().value,
14   medications = medSCT.display.value,
15   allergies = allergies.code.coding->flatten()->select(system.value="http://snomed.info/sct").display.value,
16   problems = problemList.code.coding->flatten()->select(system.value="http://snomed.info/sct").display.value,
17   observations = observations
18 }
19 IPSSummary
```

Results Explorer

Name	Class	Data
Context	Bundle	<Bundle: TRITTCone_Bundle>
patient	Patient	<Patient: TRITTCone_Patient>
allMedications[2]	Sequence(Medication)	Sequence(<Medication: TRITTCone_Medication>, <Medication: TRITTCone_Medication>)
medCodes[5]	Sequence(Coding)	Sequence(<Coding: TRITTCone_Coding>, <Coding: TRITTCone_Coding>, <Coding: TRITTCone_Coding>, <Coding: TRITTCone_Coding>, <Coding: TRITTCone_Coding>)
medMeth[1]	Sequence(Coding)	Sequence(<Coding: TRITTCone_Coding>)
medSCT[2]	Sequence(Coding)	Sequence(<Coding: TRITTCone_Coding>, <Coding: TRITTCone_Coding>)
allergies[2]	Sequence(AllergyIntolerance)	Sequence(<AllergyIntolerance: TRITTCone_AllergyIntolerance>, <AllergyIntolerance: TRITTCone_AllergyIntolerance>)
problemList[2]	Sequence(Condition)	Sequence(<Condition: TRITTCone_Condition>, <Condition: TRITTCone_Condition>)
observations[7]	Sequence(Observation)	Sequence(<Observation: TRITTCone_Observation>, <Observation: TRITTCone_Observation>, <Observation: TRITTCone_Observation>, <Observation: TRITTCone_Observation>, <Observation: TRITTCone_Observation>, <Observation: TRITTCone_Observation>, <Observation: TRITTCone_Observation>)
IPSSummary	Tuple	<Tuple>
Result	Tuple	<Tuple>
patientDetails	String	Martha Delarosa
medications[2]	Sequence(string_primitive)	Sequence(Product containing anastrozole (medicinal product), Cimofuga racemosa extract (substance))
[1]	string_primitive	Product containing anastrozole (medicinal product)
[2]	string_primitive	Cimofuga racemosa extract (substance)
allergies[1]	Sequence(string_primitive)	Sequence(Substance with penicillin structure and antibacterial mechanism of action (substance))
[1]	string_primitive	Substance with penicillin structure and antibacterial mechanism of action (substance)
problemList[2]	Sequence(string_primitive)	Sequence(Menopausal flushing (finding), Malignant tumor of breast)
[1]	string_primitive	Menopausal flushing (finding)
[2]	string_primitive	Malignant tumor of breast
observations[7]	Sequence(Observation)	Sequence(<Observation: TRITTCone_Observation>, <Observation: TRITTCone_Observation>, <Observation: TRITTCone_Observation>, <Observation: TRITTCone_Observation>, <Observation: TRITTCone_Observation>, <Observation: TRITTCone_Observation>, <Observation: TRITTCone_Observation>)
[1]	Observation	<Observation: TRITTCone_Observation>
[2]	Observation	<Observation: TRITTCone_Observation>
[3]	Observation	<Observation: TRITTCone_Observation>
[4]	Observation	<Observation: TRITTCone_Observation>
[5]	Observation	<Observation: TRITTCone_Observation>
[6]	Observation	<Observation: TRITTCone_Observation>
[7]	Observation	<Observation: TRITTCone_Observation>

and lets ask if their renal function is ok:



FHIR4.gello_model renaFailure_IPS.gello x

```
1 imports FHIR4 /* International Patient Summary */
2 Context Bundle
3
4 Let observations: Sequence(Observation) = entry.resource.Observation->flatten()
5 Let recentCreatinine: Observation
6   = observations->select(o|o.code.coding.first().code.value='14682-9')
7   ->sortedBy(effectiveDateTime.value).last()
8 Let raised : Boolean = recentCreatinine.valueQuantity.unit.value = 'umol/L'
9   and
10    recentCreatinine.valueQuantity.value > 100
11 -----
12 Tuple{
13   Creatinine = recentCreatinine.valueQuantity.value.toChar()
14     + ' ' + recentCreatinine.valueQuantity.unit.value,
15   Raised = raised,
16   Comment = If raised
17     then 'Consider chronic kidney disease.'
18     else ''
19   endif
20 }
```

Results Explorer

Name	Class	Data
Context	Bundle	<Bundle: TRTTIClone_Bundle>
observations[8]	Sequence(Observation)	Sequence{<Observation: TRTTIClone_Observa
recentCreatinine	Observation	<Observation: TRTTIClone_Observation>
raised	Boolean	true
Result	Tuple	<Tuple>
Creatinine	String	125 umol/L
Raised	Boolean	true
Comment	String	Consider chronic kidney disease.