# NLU Design Best Practices

### NLU Design Fundamental Principle: follow the Language, not the Function!

NLU systems are designed to understand linguistic patterns, rather than functionality. This means that your NLU intents and entities need to group together utterances with similar meaning, rather than utterances that end up routing to the same agent skill or that are associated with the same IVA response.

So, always group utterances under intents based on the meaning of that utterance, rather than the specific functionality of a given IVA – meaning does not change, but functionality does, so we want to make sure our intents and training utterances are future-proof. NLU systems are designed to understand.

For example, supposed that in your IVA implementation any request for canceling a reservation will go to an agent. You may be tempted to have an intent called “agent_transfer” that contains both utterances like “talk to an agent” and “cancel my reservation”, since both types of requests end up being routed to an agent. However, if in the future the customer decided to build a self-service feature around canceling a reservation, we would have to edit all the training phrases within “agent_transfer” to pull out the ones that are about canceling a reservation, causing us extra work. If we, on the other hand, base our intents on meaning from the start, we can initially route the “reservation_cancel” intent to an agent, and build a self service feature for that intent later on, sparing us the training re-work.

## Naming Conventions

Sticking to a naming convention that is shared across project and designers allows for reusability of intents and examples across projects, thus making NLU design more efficient. Below is the naming convention we adopted, try to stick with it whenever possible.

For **self-service** intents, we adopt the convention **Topic: Action**. We use “:” to separate the two components (topic, action). This convention allows us to visually group together intents that are associated with the same topic (e.g., Account). For utterances that do not contain a verb/action, we will use General as the Action component. Here are a few examples:

- Account: General this intent would be used for utterances like “account” or “my account” where the request is ambiguous, and we do not know what the user intends to do with their account.
- Account: Close: this intent would be used for utterances like “cancel my account” or “close my account
- Account Balance: Check: this intent would be used for utterances like “check my balance”, “check my account balance” or “how much is in my account”

## To Intent or To Entity, That is the Question..

One common source of inconsistency/variability on how IVAs are designed concerns the choice of using an intent vs. an entity to capture specific nuances of a user’s utterance.

In general, we should prefer an entity whenever there is large number of possible values for a given category. For example, if a request is around scheduling an appointment for a given date, we would not want to have a different intent for all the possible values of that date and we would have a “reservation_schedule” intent, together with a date entity to account for all the possible values (march 1st, June 18th, December 2nd, etc.). This applies to all cases where there is a large range of values for a category: numbers, colors, products, sizes, cities, states, etc. In these cases, always use an entity and prefer a **built-in system entity** (indicated in Dialog Flow with the prefix “sys.”), as NLU accuracy for pre-trained system entities is much higher than for custom, developer-defined, ones.

Let’s now review cases that are less clear-cut than the one above. What should we do when we have categories that can take only a handful of values, like “new” vs. “existing” or “platinum” vs. “gold” vs. “silver”? In these cases, the recommendation is that if the range of values of the category you are working on does not have a clear set of values that can be succinctly expressed by a predictable set of phrases, an intent should be preferred.

## Training Utterances & Other Tips

- Aim for at least 20 utterances per intent
- Make sure your examples are varied
- Keep the length of the training examples in the range of 1-15 words
- Ensure that the distribution of your utterances is balanced. You don’t want some intents to have hundreds of training examples while other have only a few
- Typically, you will want to avoid typos and STT misrecognitions in your training utterances. Ideally, consistent STT misrecognitions would be corrected before an utterance is sent to NLU (by adding phrases hints, or using custom acoustic models). If this is not possible, and you have some very consistent STT misrecognitions that are causing NLU failures, then it is ok to add them to your NLU training, but be aware of potential side effects on your NLU classification accuracy.

## NLU Metrics

See definitions and examples here

- **Precision**: the rate at which the model is correct when it makes a prediction. For example, if when the model says that a group of utterances belong to the intent “reservation_cancel”, the model is correct 90% of the time, .9 is the model’s precision for that intent. We typically want this value above .85 (but depending on the use case, we may want to bias for higher precision or higher recall)
- **Recall**: the rate at which the model is correct at identifying all the examples that belong to a given category. For example, if a model correctly categorizes as “reservation_cancel” 80% of the utterances that are about canceling a reservation, the model’s recall for that intent is .8. Imagine a scenario in which 90% of the cases belong to category 1 and 10% to category 2. A model that picks category 1 100% of the time, would have perfect recall for category 1 (it finds all the category 1 cases) but low precision (when it says category 1, it is wrong 10% of the time). We typically want this value above .85 (but depending on the use case, we may want to bias for higher precision or higher recall)
- **F1**: harmonic mean of precision and recall, so it’s a composite measure that takes into account both. We typically want this value above .85 (but depending on the use case, we may want to bias for higher precision or higher recall)
- **Coverage**: The proportion of a data set for which a model makes a prediction. For example, if we have an IVA with only 3 intents, we could have great precision and recall for our 3 intents but be missing out on most topics that our users want to talk about. Typically, we want this value to be above .75