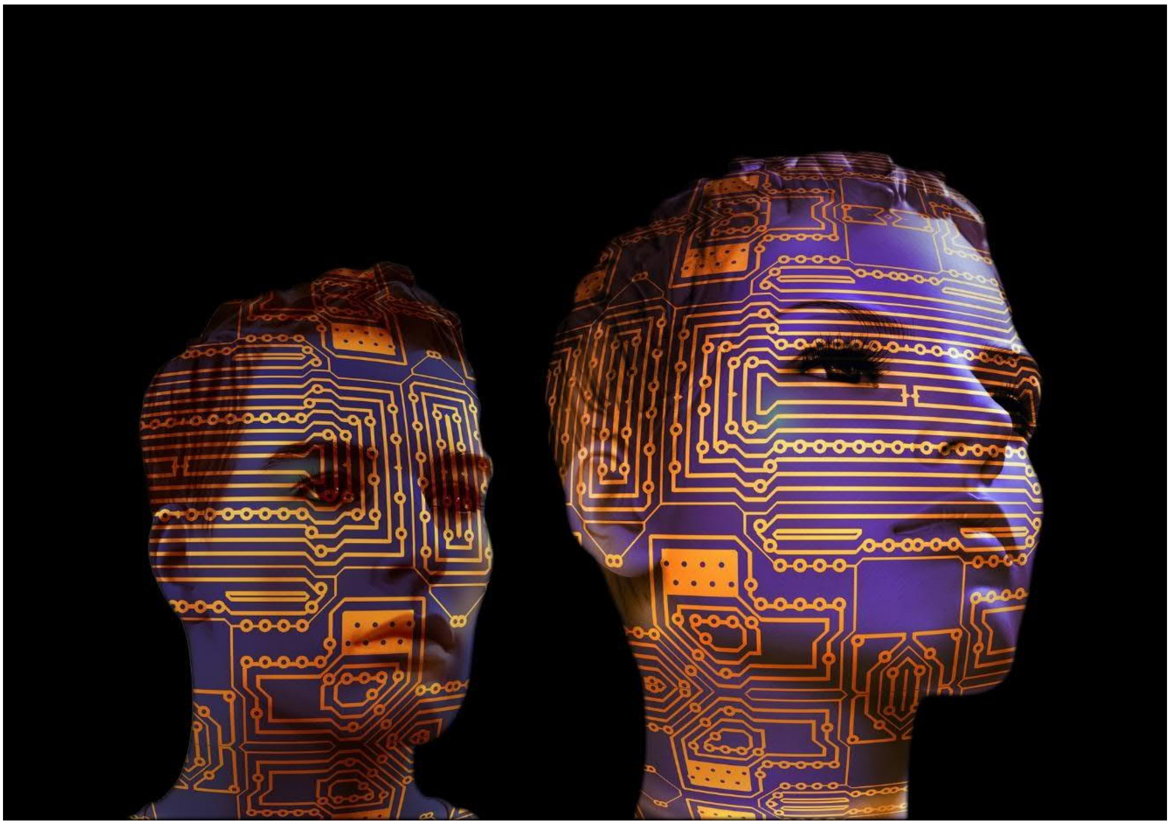
TESTING VOICE USER INTERFACES

Overview of metrics, methods and automatization



Abstract

There are some challenges during the test of voice user interfaces. For one thing, it is not clear how to measure the accuracy of voice user interfaces. On the other hand, it's hard to find adequate spoken utterances to be used in the test. The question: How can we automatize tests of voice user interfaces is at the agenda periodically. This text takes a quick glance behind the scenes of the test of voice user interfaces.

Created by

Diethelm Dahms, Speech & Phone GmbH, translated from German, originally published at https://www.testing-board.com/testmanagement-von-sprachbenutzerschnittstellen-vui/

1 TABLE OF CONTENTS

2 QUALITY ASSURANCE

2.1 Tools usable for quality assurance within the agile software development

Even agile development teams stand in the need of requirements, since successful software tests increase the software quality, but create above all confidence. Therefor these descriptions and artifacts belong to agile software development:

- Definition of Ready – conditions for starting the task
- Definition of Done – conditions for finishing the task
- User stories – description of the task's content
- Scenarios – points to take in account for the task
- Acceptance criteria – conditions for acceptance of the task

These documents outline incrementally the conditions to realize a concrete task. If certain documents are absent, there are missing preconditions to successfully perform or prove a certain task.

2.2 From the user story to a test case

The team describes the software requirements from the customers point of view and notes them in user stories. Even acceptance criteria and the definition of done are components of requirements.

The definition of ready is a good source for latter developers or testers, where to get knowledge on the task. The definition of done is an excellent source to phrase test cases. Starting from acceptance criteria and the scenarios testers are enabled to describe several test cases.

2.3 Example – test of a registration

The user story describes a registration process an included registration form. The following details are inquired during the registration of users

- name (mandatory)
- email address (mandatory)

- number of id document (mandatory)
- salutation (optional) and
- first name (optional)

The test scenario expects, that missing entries in mandatory fields are proved individually and combined with other mandatory fields. The missing entry of optional fields are proved as a whole. Doing so, a test case occurs containing nine test steps including the special or combined missing of the mandatory fields name, email address, number of id document and two further test steps for missing optional fields salutation and first name. This test approach does not provide a complete test coverage.

It is irrelevant according to the validity of the test execution, how letters are entered. The speed of typing is irrelevant for the correctness of optional or mandatory fields using graphical entering.

There are several graphical dialog elements (menus, lists, buttons) to ease the handling, to channel the interaction and to decrease wrong entries. A menu can be clicked, tipped, swiped or chosen by short key.

Voice user interfaces have only few standard elements. The input of natural voice is felt as freedom. The simplification of interaction possibilities is felt as a limitation of input.

Options in voice user interfaces increase mental strain, because there is a back channel using language only. Acoustic feedback is not to recap as easy as graphical dialog elements and decrease the acceptance and the usability of the interface. This dilemma leads to a situation, that dialogs must be offered in units that enable both a little stress and a big variability of utterances.

2.4 Example – linguistic variation in phrases

If the registration form from the previous example is filled in by voice, there is an additional uncertainty concerning the entry options. Linguistic variances have be considered for each entry field. In addition, there should be the possibility to indicate all fields in one utterance. The following analysis of existing and missing fields must prove if mandatory or optional fields are missing and offer the possibility of correcting. This dialog step is to be considered in the test scenario.

Just linguistic variants include variations of verbalisation. The usage of adverbial phrases, figures of speech, utterance of needs or the use of ambiguous terms. For these the following table shows variants for the field name only.

- My name is Paula.
- I am Paula.
- Others say Paula to me.

- Paula Miller.
- Miller, Paula.
- Call me Paula.

The following example shows how utterances of the registration process could be expressed, which variations could appear and what is be expected of application's reaction.

Field of registration	Missing fields and uncertainties
My name is Gerard Walter. The address is Gerard.walter@email.xzy, ID number GWZYX786235698.	The salutation is missing, but it is not expected to be said. There is a vagueness concerning first and family name. This and the missing salutation must be clarified in the dialog.
The ID number is G-W-Z-Y-X-7-8-6-23-56-98. I am Walter and the email is gerard.walter@email.xzy	The sequence of the fields is freely uttered and the system must assign them correctly. The uttered name is ambiguous and can't be assigned correctly to first or family name. Therefore, the possibility of the implicit assignment of the (optional) salutation fails.

2.4.1 Note

During the validation of several fields in a acoustic utterance, there is a challenge showing why most applications have difficulties while recognizing multiple parts of phrases and the most applications architects omit multiple details in one utterance.

2.5 Once doesn't count – test execution in voice user interfaces

Graphical user interfaces (GUI) act independent from the manner of text entering as mentioned above. There is no difference of the entered content concerning the speed. The typing speed does not impact the correctness of most field contents as even the number of typing breaks.

However, there are differences in manner of data entry if using the voice user interface. The speed of speaking has an influence on accuracy of the speech recognition just like the volume level, the articulation or the pitch. If words are normatively not correct or they are not considered in the acoustic language model, the acoustics will be changed into text insufficiently. This can decide on success or fail of the entry field or the application.

Indeed, there are typing failures during text entry, but in my opinion, they are less heavily felt as the difference in speaking. Typing mistakes are more easily blamed to user than to the application.

Some certain speech recognizers are still having difficulties to recognize acoustic specialties of dialects after many years of research, development and implementation.

This behaviour shows an essential difference between the test of graphical and voice user interfaces: According to a GUI the number of test cases designed for a specific dialog is determined by content. Differences of spelling, the inclusion of regionally used special characters (umlauts, accents, diacritic characters) increase even the number of test cases, but it is still sufficient to execute each test case exactly once, to be able to decide, if a test case has been passed or failed.

According to voice user interfaces single testers are able to reflect the diversity of several utterances only in syntactical regard. The variance of articulation, intonation, the pitch and the dialects cannot be reflected by single testers. Therefore, it is necessary to establish a test team, in order to execute the same test cases by different persons.

The test results must be summarized according to the several test executions. There are the mentioned cause variables pitch, speaking velocity, wording, environment noises, regional origin. These additional cause variables multiply the test effort enormously.

Even from the diversity of test executions, there is no clear test result, but a likelihood, if a test case has passed or not. By this, a basis of trust arises, but not a absolute certainty as in graphical user interfaces.

3 COMPILE TEST UTTERANCES

3.1 Include linguistic competence and style for test data

The mentioned cause variables influence the compilation of test data. Preferably, phrases of test data should reflect all cause variables representatively. There is still a challenge that the distribution of cause variables in the target group is generally unknown. Since the weighting of the observed parameters differ in test and field population, differences in recognition precision are to be expected. For choosing acoustic test date these cause variables are considered mandatory.

- Gender
- Regional origin
- Age
- Educational level

Considering these criteria enables the participation of all population groups to the speech application. Some of these parameters can be defined before the collection of test data, while others can be defined, if recordings are available. If the desired distribution of parameters is not met, additional recordings are necessary, to represent the expected distribution of influence variables.

As a further aspect of test utterances is the linguistic style of the utterances. The following kinds are shown within linguistic scriptures (Sprachstilarten - Dein Sprachcoach, Standardsprache und Variation - Dürscheid / Schneider, Die Stimme, wie sie wirkt und was sie über uns verrät - Spektrum der Wissenschaft, German).

- youth language
- language according the formation
- everyday language
- dialects
- formal language
- standard language
- social origin
- native speaker or acquired language

The influencing variables of the particular parameters are differently-sized according to planned application and the effort to reproduce the different linguistic styles is varying. In addition,

3.2 Approach to the creation of test utterances

The execution of software tests is as good as the open mind of the participated persons. The aim of software tests is the proof of software quality. That's the reason of the principle "Tests create trust." During the test execution it is also realized, which differences are faced by the development team in order to ensure a requested degree of software quality. Even for members of the development team are biased members the test team has this risk. Therefore further parameters must be taken into account for the collection of test data, so that the perception of the test team is not reflected in test utterances.

On the whole, there are the following possibilities for collecting and noting the test utterances. The pros and cons of the approaches will be described afterwards.

- Laboratory approach
- Fieldwork approach
- Guidelines approach

Before showing the discrete approaches of collecting test utterances, it will be shown, which possibilities of linguistic variance may exist and which variables in test utterances must be considered, to get a reasonable basis of trust.

The experience shows, that systems proven by homogenous test groups have difficulties in the field use. That's why, heterogenous test groups to prove applications is a good deal.

3.2.1 Examples for different test utterances with identical content

The following table shows possibilities for different phrases of utterances with identical content. On the one hand there is the manner to express issues linguistically. On the other hand, there is the possibility to change phrases with articulation. People from the North speak differently than people from the South, and certain linguistic experiences create other forms of expression.

Cause variable	Variant type	Amount	Example
Syntax	Simple direct question	1-2	What's the time?
	Simple indirect question	1-2	Could you tell me, what the time is it?

	Simple request	1-2	Tel me, what's the time?
	Simple ask or need	1-2	I would like to know what's the time.
	Complex requests including several adverbial phrases	2-3	What's the time in Tokyo?
Articulation	Speaking velocity	2-3	Fast, slow, hesitations, interrupts
	Pitch	3	High, low, middle
Linguistic Style	Sociolect	2-5	Slang, technical language, youth language, standard language
	Dialect (Regiolect)	3-5	According to existing regional variants of the language

This bunch of variations is valid per language of an application. If an application will be implemented in different languages, these variants have to be built per language. To achieve a complete coverage of these variables the concerned cause variables must be multiplied. This leads to the formula:

Number of test utterances per test case =
syntax variants
* of articulation
* number of linguistic styles to be considered

The use of the application under test defines the number of linguistic cause variables to be considered. If applications are planned for technical languages only, certain linguistic styles can be dissolved out of the consideration. The more general of the applications use cases, the bigger is the reflecting variance of linguistic styles during the test.

3.2.2 Define heterogenous test utterances in the lab

If following the laboratory approach, the test team defines the test utterances and observes the compliance of the required parameters. Sind test teams have a modest size, the linguistic variance of test utterances is hardly to create.

The essential influence of the creation of variance is the heterogeneity of the test team. While syntactic influence can be easily created and included by eloquent human, the variances concerning the linguistic style and articulation can rarely be created.

If recording utterances in articulated regard, it could be tried to create phrases with syntactic diversity in the lab and record them by different people. Doing so you will win articulatory diversity. The disadvantage is, that spoken language is rarely created in the laboratory, but written language. Written language is only partially usable for test sentences in voice user interfaces.

> **A talk is once and for all not a write.**
> **Friedrich Theodor Vischer**
> https://gutezitate.com/zitat/169168 (de)

3.2.3 Field research for creation of heterogenous test utterances

The collection of test utterances according to articulation, linguistic style and competence can easily be done by field research. At the same time, the effort within the test team is changing. The lab approach requires competence of linguistics, but the field research requires statistics, organisation etc. By using this approach, the effort shifts to the description of the application use, because this knowledge is not available outside the project team. This knowledge and the ability to vividly describe it for the recruitment of participants is necessary.

Organizational work is required, to manage the number of recordings. The syntactical diversity ideally results by itself, but it must be verified. At the same time, there will be outliers, there is test data, which must be discarded.

> **One examines the parts only, to judge the whole, one examines all reasons, to realize all effects. - - Charles de Secondat, Baron de Montesquieu,**
> https://www.aphorismen.de/zitat/123974 (de)

3.2.4 Get test utterances by test scenarios

The definition of test scenarios is a further form of test data collection. Doing so, the syntactic diversity ist not created by a test team as described above, but situations are described to be imagined by probands and may phrase freely, what the speak.

The approach with the usage of test utterances differs from the approach with the more general description of a testing situation. Instead of using a rigid definition of a test utterance, a test situation is described and the test team rely on the creativity of test persons.

In this approach the biggest challenge is it to not induct probands just to repeat phrases. According to the competence of probands the ability to imagine is varying. There are limitations of the description of a situation without using specific word. Overview

> **The quality of a researcher is not measured by the number of answers, but by the quality of questions asked.**
> -© Aba Assa, https://www.aphorismen.de/zitat/154151 (de)

3.2.5 Postprocessing of test utterances

The effort of getting test data differs according to the chosen approach. This table shows the estimation of the different forms of getting test data.

Parameter	Effort scenarios	Effort field research	Effort Laboratory
Syntax	High	Difficult	Simple – middle
Articulation	Simple	Simple	Very high
Linguistic style	Simple	Simple	Very high

Language	High	High	Very high

No matter what kind of getting test utterances, there are postprocessing tasks. The obtained recordings must be monitored. During this task, they must be transcribed, the meta data of a test utterance as language, age, articulation etc. must be noted and the desired recognition results must be captured according to the application. This effort multiplies for each gathered language.

The documentation of meta data assures, how the required test parameter regarding syntax or articulation are met. The documentation of the contents must be in a form, that test results easily can be compared with requirements latterly. The following table shows by the test case “provide time”. It is assumed that the recognition of a time can be improved. The time is stored in the both transfer parameters hour and minute. (The test case expected and actual results are literally translated from German and can’t be used in English, note of the translator.)

Test utterance	Style, region	Social	Expected	Actual	Result
Sixteen hours fifteen	Formal North	M30 Middle	Hour: 16 Minute: 15	Hour: 16 Minute: 15	PASSED
Quarter five afternoon	Slang South	F40 High	Hour: 16 Minute: 15	Hour: 5 Minute: 15	FAILED
Quarter past four in the afternoon	Slang West	M50 Uni	Hour: 16 Minute: 15	Hour: 4 Minute: 15	FAILED
Quarte past four afternoon	Slang East	F20 Occupation	Hour: 16 Minute: 15	Hour: 4 Minute: 15	FAILED
Should be shortly after four during	Slang North	M10 Pupil	Hour: 16 Minute: 15	Hour: 4 Minute: 0	FAILED

today’s afternoon					
In two hours	Familiary South	F30 Middle	Relative time	Hour: 18 Minute: 0	FAILED
Ten to four at noon	Slang East	F40 Occupation	Hour: 15 Minute: 50	Hour: 4 Minute: 50	FAILED

- Style: Formal, Slang, Familiary, Elevated, Standard
- Region: North, South, East, West
- Social, Sex, (M, F, D) , decade of age, educational qualification (middle School graduation, high school, university, occupation, pupil)

3.3 Choose test methods for speech recognizers

3.3.1 Static test methods

Before talking about, how test utterances can be played to an application, it must be clarified whether dynamical or static test procedures are to be used. In my point of view there is a clear preference to dynamic test methods, because the result of the application can only be shown by a recognition that had taken place. Furthermore, it is to clarify, whether the dynamic test must be executed in the whole application or whether ty dynamic test of grammars or recognition package is sufficient. Nevertheless, static test procedures must not be excluded.

Static test methods can be used for speech recognitions. Doing so, the process is to be proved, that leads from requirements to the recognition package. It is to be clarified if the used artifacts are under version control and if components can be rolled back if necessary. In addition, source code files, used for the creation of speech recognition modules are syntactically correct. This correctness must be proved for the computer language – mostly a derivative of XML – and of the natural language. The dictionaries of pronunciation for speech recognition and speech output must be included in this correctness test. A relevant possibility – a linguistic generation test – is just a static test execution. Starting from the compiled package of the speech recognizer, a list of phrases is generated, the grammar is able to recognize. By this utterances can be found which do not correspond to the linguistic conventions.

3.3.2 Example – Speech recognition grammar

This grammar is able to order Italian pastry, it can be found at https://vivoka.com/speech-recognition-grammar-editor-plugin/. The grammar is simplified for clarity reasons, but it is modularized, but on the other hand fairly readable.

```
<main>: <verb> !repeat((a | <number>) <pizza> [and [<verb>]], 1, *);
<verb>: I (want | would like);
<number>: !tag(NUMBER, 1 | 2 | 3);
<pizza>: [pizza] !tag(PIZZA_TYPE, margherita | proscuiutto e funghi |
capricciosa
   |  vegetariana | calzone);
```

That's why it is good to prove grammars at textual level. The following phrases can be recognized by this grammar.

```
I want a margherita
I would like a pizza capricciosa and 2 vegetariana
I want a margherita, 2 capricciosa and I would like a calzone
```

During this half static test, it would stand out, that the following points should be added for a productive grammar.

```
all phrases must start with „I“ -> it should be possible to use „we“
polite phrases are not possible -> „please“ to the start or the end should be
possible
the number of opening phrases is very limited -> „bring me“, „make me“ and so
on should be included
```

3.3.3 Dynamic test procedures for speech recognition

There are several possibilities of dynamic test procedures for established speech recognitions. At the one hand, you can execute using transliterated input. Doing so text files carrying different formulations are used for the prove of recognition models. The recognition results show afterwards, which hypothesis the speech recognizer has used. By this approach deviations at syntactical level can be found finer and proved on a regularly base.

And speech recognizer can be proved interactively with single phrases. This is starting point of a more empirical procedure during the test. This dynamical test method is often used during the development, but is usable during the test, too. However, it comes along with high manual effort.

Speech recognizer can be fed with recordings, too. There is the possibility to use them interactively or automatically to hand over them with bigger corpora of a specific instance of a speech recognition package and to monitor the recognition results. Meaningful test metrics arise by starting from the test corpus and the therein contained expectations of a test utterance, this test automatization is convenient to be used in regression tests, to prove a reached quality level.

All mentioned test methods for speech recognizers can be done without the end device and interactively or automatically. The static tests are done as a review for documentation or source code, the dynamical tests are executed at the platform.

3.3.4 End-to-end-tests of speech recognizers

Finally, dynamic test procedures can be executed even at the end device. For these tasks, you have to keep in mind a very high manual effort, because the handling of end devices like phone, smartphone, speaker, infotainment systems can be done by voice, but the gathering of the recognized parameters at the system is not easy. In this case, the manual execution is necessary, because the automatic test execution brings up follow-up failures, if dialog steps are not coordinated and return consequently to a starting point.

The mentioned test procedures relate to speech recognizer in a narrower sense. That’s the tailored speech package of the application. Additional monitoring points of end-to-end-tests are speech output, dialog ability and the connection to backend systems.

It must be assumed that a complete test coverage of this test procedure is not available, since modern voice user interfaces have to much input and output speech parameters. That’s why end-to-end-tests are to be prioritized according the requirements.

3.3.5 Non-functional tests of voice user interfaces

A further required test procedure are non-functional tests. The first component under test is the performance and load behaviour of speech recognition and speech output in client-server-configurations. The storage of speech recordings or other recognition results must be proved, too.

The ruggedness of the speech recognition concerning background noise, unusual articulation are to be planned on a regular base. These tests can be done with selected utterances for each release automatically.

4 TEST EXECUTION

4.1 Reproducibility in the test execution

It is important to show a reproducible system behaviour during test execution. Voice user interfaces and graphical user interfaces differ enormously in this aspect. That's why the test of speech recognitions has to be done with speech recordings ideally. This approach only increases the possibility to reproduce the system behaviour. If using speech recordings for the test execution the triggering event is identically in all repetitions of the test execution and can reproduce the system behaviour.

If the possibility to play speech recordings is not available, the text of test utterances used in the test case is to document at least and in addition the same persons must be commited in the test execution.

4.2Automatization of test execution

If the speech recognizer offers the possibility, to use text files or audio recordings for recognition plentiful, you should make use of it absolutely. For this purpose, an essential and crucial prerequisite is the development of a test corpus. However, the maintenance of the test corpus increases the test effort totally. The test corpus is to be maintained that the individual revisions can fit to the software release of the tested application.

Concerning automatization, the effort of configuration of automatization, the execution and the reporting should be less than the manual preparation and execution. To show a reached software quality an automatic test can be used, according new functions manual tests are the means of choice. As in every test, the permanent repetition of existing tests prevents the finding of new failures.

Each test phase includes the audit of defect handling, new features and existing features. This results in the following test scope for each software release

- fault repair test
- feature test
- regression test

Fault Repair tests feature tests have a low potential of automatization, only. Test automatization is to be planned for regression tests. To omit test fatigue, regression tests must be updated and to be changed regularly.

5 TEST REPORTING

The reporting of test execution of voice user interfaces differs insignificantly from usual test reporting. The expected and the actual behaviour are contrasted. There is a challenge in the evaluation of the recognized hypotheses of the speech recognizer. The recognition accuracy of the phrases in the test corpus must be shown according to the intention. The intention must have been described during the creation of the test corpus.

The evaluation of the speech recognition results is based on statistic details. That means, there is a certain probability for a concrete test utterance from test corpus, that this sentence is interpreted as expected. This detail differs from graphical user interfaces, which result in passed or failed. The possibility of correct recognition must be shown across the used test parameters.

If the test has not been executed at the end device, there is the need of a qualified description of the transferability of the test results in lab conditions to the conditions at the end device.

6 RESUMING THE TEST PROCESS

The test process of voice user interfaces differs at some points from the test of graphical user interfaces (GUI). This is caused by the statistic distribution of test results, which are less deterministic than tests of GUI.

These several linguistic influences must be considered before and during the test data acquisition, to ensure that the distribution of cause reasons to test data relates to the implied expectations in the target system. Test data can be collected in laboratory, by free or leaded field studies.

Speech recordings should be used preferrable during the test execution, to get reproducible test results.

The test reporting must use metrics to show the dimension of possibility of the recognition according to the expected results per test parameter.

7 ADDENDUM

7.1 Changes

Date	Version	Author	State	Changes
7.6.22	0.5 en	Dd	Finished	Updated document template
9.8.22	0.6 en	Dd	In Progress	

7.2 Table of Images